Spectral Graph Theory Notes

Instructor: Gary L. Miller
Notes compiled by: Lichen Zhang
Carnegie Mellon University
15-859N Spectral Graph Theory

Spring 2020

Contents

1 Introduction and Course Topics **7**
1.1 Spectral graph theory . . . 7
1.2 Graph sparsifiers . . . 8
1.3 Topics we hope to cover . . . 8

2 Graph Laplacian, Effective Resistance, Energy **10**
2.1 Yet another way to search a graph . . . 10
2.2 Resistive model of a graph and random walks . . . 10
2.3 Resistance theory . . . 11
2.4 Effective resistance and conductance . . . 12
2.5 Computing effective resistance from a to b . . . 13
2.6 Another way to view the Laplacian . . . 15

3 Energy and Rayleigh's Monotonicity Law **18**
3.1 Current and energy (power dissipation) . . . 18
3.2 Two types of flow . . . 19
3.3 Thomson's principle . . . 20
3.4 Rayleigh's monotonicity law . . . 21

4 Random Walks on Graphs and Commute Time **23**
4.1 Random walks on graphs . . . 23
4.2 A small example . . . 23
4.3 Large graphs . . . 24
4.4 Two views of a random walk . . . 24
4.5 Important parameters . . . 24
4.6 Random walks: the symmetric case . . . 25
4.7 The pinball machine problem . . . 25
4.8 A recurrence for h_x . . . 26
4.9 The electrical version . . . 27
4.10 Back to the path example . . . 27
4.11 Interpretation of current for random walk . . . 28

4.12 How to compute hitting time . . . 30
4.13 Commute time . . . 31
4.14 Another example . . . 32

5 Random Walks and Mixing Times 33
5.1 Random walk as a matrix-vector product . . . 33
5.2 Stationary distributions and convergence . . . 34
5.3 Symmetrizing the walk . . . 35
5.4 Spectral theorem . . . 36
5.5 Perron–Frobenius theorem . . . 36
5.6 Back to the symmetric case . . . 37
5.7 ℓ_1 norm and Cauchy–Schwarz . . . 38
5.8 Proof of the spectral theorem . . . 39
5.9 Random walks on the path graph . . . 40

6 Perron–Frobenius and Symmetric Perron–Frobenius 41
6.1 Symmetric Perron–Frobenius . . . 41

7 Differential Equations, Matrix Exponentials and Laplacians 45
7.1 Differential equations on graphs . . . 45
7.2 Ultra lazy random walks . . . 46
7.3 Springs and graph Laplacians . . . 47
7.4 The normalized Laplacian and spring-mass systems . . . 49

8 Graph Sparsifiers and the Ahlswede–Winter Theorem 50
8.1 Graph sparsifiers . . . 50
8.2 Generating a sparse subgraph by edge sampling . . . 50
8.3 Chernoff bounds . . . 51
8.4 Ahlswede–Winter theorem . . . 51
8.5 Picking edges by effective resistance . . . 53
8.6 Upper estimates of effective resistances . . . 54

9 Bounding Eigenvalues, Courant–Fischer, Path Embedding 56
9.1 Mediant of fractions . . . 56
9.2 Courant–Fischer (CF) . . . 56
9.3 Degrees and the extreme eigenvalues . . . 57
9.4 The Loewner order . . . 58
9.5 Pseudo-inverse . . . 59
9.6 Effective resistance and the Loewner order . . . 59
9.7 Upper bounding λ_2 . . . 60
9.8 Lower bounding λ_2 . . . 60
9.9 Path embedding . . . 61
9.10 Laplacian as a sum of Laplacians . . . 62
9.11 Back to λ_2 for P_n . . . 63

10 Golden–Thompson Inequality via Weyl's Majorant Theorem **64**
10.1 Golden–Thompson inequality . . . 64
10.2 Real linear algebra needs complex numbers . . . 64
10.3 Weyl's majorant theorem . . . 66
10.4 Lie product formula . . . 69
10.5 Back to the proof of Golden–Thompson . . . 70

11 Matrix Chernoff Bounds (Ahlswede–Winter, Tropp) **72**
11.1 Chernoff bounds and matrix Chernoff bounds . . . 72
11.2 Tropp implies Ahlswede–Winter . . . 74
11.3 Examples . . . 75

12 Graph Cuts and Eigenvalues: Cheeger Inequality **77**
12.1 Graph cuts and eigenvalues . . . 77
12.2 Cut measures . . . 77
12.3 Main Cheeger theorems . . . 79
12.4 The easy direction for the quotient cut . . . 79
12.5 The upper bound for the conductance . . . 80
12.6 The proof: preliminaries . . . 82
12.7 The proof . . . 83

13 Direct Linear Solvers, Nested Dissection for Planar Graphs **86**
13.1 Direct linear system solvers . . . 86
13.2 Computing an LU decomposition . . . 87
13.3 Symmetric positive definite matrices . . . 89
13.4 Schur complement . . . 89
13.5 Graph theoretic view of pivoting . . . 90
13.6 Perfect elimination . . . 92
13.7 Chordal graphs . . . 93
13.8 Pivot strategies for the square mesh . . . 93
13.9 Nested dissection . . . 94
13.10A view of fill for planar graphs . . . 95
13.11Fill and work for nested dissection . . . 96
13.12Block Gaussian elimination . . . 98

14 Basic Iterative Method, Extrapolation, Chebyshev Acceleration **100**
14.1 The basic iterative method . . . 100
14.2 Richardson's method . . . 101
14.3 The error for Richardson . . . 101
14.4 The extrapolated method . . . 102
14.5 Polynomial acceleration . . . 104
14.6 Chebyshev polynomials and acceleration . . . 105
14.7 Chebyshev plus extrapolated method . . . 106
14.8 From polynomial recurrence to iterative algorithm . . . 108

15 Conjugate Gradient Method and Steepest Descent **109**
15.1 Linear systems as minimization problems . . . 109
15.2 Crash course on matrix calculus . . . 109
15.3 Richardson's algorithm and a variable step size . . . 110
15.4 Steepest descent . . . 111
15.5 Convergence rate . . . 111
15.6 Steepest descent: an example . . . 112

16 Conjugate Gradient Continued **113**
16.1 Krylov subspaces . . . 113
16.2 Conjugate gradient . . . 114
16.3 The one-matrix-vector form of CG . . . 115
16.4 CG and polynomial approximation . . . 116

17 Preconditioned CG and Low-Stretch Spanning Trees **119**
17.1 Preconditioned iterative methods . . . 119
17.2 Examples of preconditioners . . . 120
17.3 Convergence rates per iteration . . . 120
17.4 Estimating $\kappa(B^{-1}A)$ . . . 121
17.5 Subgraph preconditioners . . . 122
17.6 Spanning tree preconditioners . . . 122
17.7 Finding a better spanning tree . . . 124
17.8 The trace and effective resistances . . . 125
17.9 Back to C_n with tree P_n . . . 126

18 Preconditioned CG and LSST Continued **127**
18.1 Low stretch spanning trees (LSST) . . . 127
18.2 Polynomials that are small at all but a few eigenvalues . . . 127
18.3 PCG with a low-stretch spanning tree . . . 128

19 Fiedler's Theorem (Nodal Domains) and Generalized Laplacians **130**
19.1 Fiedler's theorem on nodal domains . . . 130
19.2 Nodal domains . . . 130
19.3 Generalized Laplacians . . . 132
19.4 Cauchy's interlacing theorem . . . 133
19.5 Fiedler's theorem for generalized Laplacians . . . 133
19.6 Back to nodal domains . . . 135

20 Eigenvalues and Eigenvectors by Iterative Methods (Power and Inverse Iteration) **137**
20.1 Iterative methods for eigenvalues . . . 137
20.2 Analysis . . . 137
20.3 Inverse iteration . . . 139
20.4 Finding all the eigenvalues and eigenvectors . . . 141

21 Graph Maximum Cut via Spectral Methods (Trevisan) 142
21.1 Spectral maximum cut . . . 142
21.2 Max-Cut algorithm . . . 145
21.3 Proof of Trevisan's theorem . . . 145
21.4 Analysis of the algorithm . . . 147

22 Solving Graph Laplacians in Nearly $O(m \log n)$ Time 149
22.1 Nearly $O(m \log n)$ Laplacian solver . . . 149
22.2 High-level view of the algorithm . . . 150
22.3 The solver has two distinct phases . . . 150
22.4 Sampling nontree edges . . . 151
22.5 Graph algorithms on trees . . . 152
22.6 Timing analysis of Phase 1 assuming Phase 2 . . . 152
22.7 The critical trick to get a one-log solver . . . 153
22.8 Phase 2 . . . 153
22.9 Analysis of Phase 2 . . . 153
22.10 In summary . . . 154
22.11 Removing degree one and two nodes using Gaussian elimination . . . 155

23 Components of the Fast Laplacian Solver (Solver Chain) 156
23.1 Components of a fast Laplacian solver . . . 156
23.2 Ultra sparsifiers . . . 157
23.3 The critical trick to get a one-log solver . . . 158
23.4 Phase II of the solver chain construction . . . 158
23.5 Removing degree one and two nodes using Gaussian elimination . . . 159
23.6 We must check that the spanning tree is low stretch . . . 159

24 Random Walks with Restarts, Local PageRank, Spilling Paint 161
24.1 Personal PageRank and spilling paint . . . 161
24.2 Two random walks . . . 161
24.3 Spectral theorem . . . 162
24.4 Spilling paint view of LPR . . . 163
24.5 Pushing wet paint . . . 164
24.6 Understanding the stationary P_v for walks with resets from v . . . 165
24.7 The stationary as a flow (circulation) . . . 165
24.8 Main theorem . . . 166
24.9 Analyzing LPR using Cheeger's inequality . . . 168

25 Solving Symmetric Diagonally Dominant (SDD) Systems 170
25.1 Solving symmetric diagonally dominant systems . . . 170
25.2 Double cover . . . 172

26 Counting Spanning Trees (Matrix-Tree Theorem) 175
26.1 Random spanning tree . . . 175
26.2 Main theorem . . . 177
26.3 The undirected case . . . 178

27 Markov Chain Tree Theorem **180**
27.1 Random spanning tree via random walks . . . 180
27.2 Random walks 101 . . . 180
27.3 Markov chain tree theorem . . . 181

28 Generating Random Spanning Trees via Random Walks **185**
28.1 Forward trees . . . 185
28.2 Uniform random spanning trees from the cover time . . . 186

29 Random Walks and Perfect Bipartite Matching **188**
29.1 Randomized perfect bipartite matching . . . 188
29.2 The expected time of the random walks . . . 189

30 Guest: Machine Learning Basics for Graph Neural Networks **193**
30.1 Machine learning and graph neural networks . . . 193
30.2 Background on machine learning . . . 193
30.3 Gradient descent . . . 195

31 Guest: Backpropagation, Generalization, Regularization, ML on Graphs **198**
31.1 Backpropagation . . . 198
31.2 Generalization . . . 199
31.3 Regularization . . . 200
31.4 ML on graphs . . . 200

32 Guest: Graph Neural Network Models (GCN) **202**
32.1 Graph neural networks . . . 202
32.2 Experiments . . . 205

33 Maximum Flow via Electrical Flow **207**
33.1 Max flow via electric flow . . . 207

34 Nesterov's Gradient Method and the LRS Approximate Max-Flow Algorithm **212**
34.1 Nesterov's gradient method . . . 212
34.2 Nesterov's algorithm . . . 213
34.3 The LRS approximate max flow algorithm [LRS13] . . . 214
34.4 Main theorem . . . 214
34.5 Drain . . . 216

References **217**

1 Introduction and Course Topics

Lecture of January 13, 2020. Based on Gary L. Miller's handwritten notes.

This lecture explains what spectral graph theory is and why it is worth studying. The subject brings together graph theory, linear algebra and numerical computing, and this combination has produced faster algorithms for basic computational problems. We describe two such problems, shortest paths with negative edge weights and least-squares regression, introduce graph sparsifiers, and list the topics of the course.

1.1 Spectral graph theory

Spectral graph theory is the interplay between three areas:

1) graph theory,

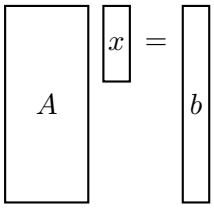
2) linear algebra,

3) numerical computing.

Its main payoff is faster algorithms, and we give two examples.

The first example is the single-source shortest path problem in a directed graph with m edges and n vertices whose edge weights may be negative. The Bellman–Ford algorithm solves it in $O(mn)$ time. In 2016, a new method [CMSV17] solved it in $\widetilde{O}(m^{10/7} \log W)$ time when the edge weights are integers of absolute value at most W; here $\widetilde{O}$ hides polylogarithmic factors.

The second example is regression. In its simplest form, we are given an overconstrained linear system $Ax = b$, where A is an $m \times n$ matrix with many more rows than columns and $b \in \mathbb{R}^m$:



Such a system usually has no exact solution, so our goal is to find a vector x that attains

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2.$$

The standard answer is to solve the normal equations

$$A^\top A x = A^\top b.$$

The difficulty is cost: computing $A^\top A$ alone takes $O(mn^2)$ time. Newer methods instead form $B = SA$, where the matrix S selects a few rows of A and rescales them, so that B has far fewer rows than A, and solve

$$B^\top B x' = B^\top S b.$$

For a well-chosen S and a given $0 < \varepsilon < 1$, the solution x' is nearly optimal,

$$\|Ax' - b\|_2 \leq (1 + \varepsilon) \min_{x \in \mathbb{R}^n} \|Ax - b\|_2,$$

and the whole computation takes less than $O(mn^2)$ time. Choosing S uses leverage scores, which we introduce later in the course.

1.2 Graph sparsifiers

For a graph $G = (V, E)$ and a set $S \subseteq V$, write $\bar{S} = V \setminus S$ and let $\mathrm{Cut}_G(S, \bar{S})$ be the number of edges of G with one end in S and the other in $\bar{S}$. For a weighted graph H, let $\mathrm{Cut}_H(S, \bar{S})$ be the total weight of such edges.

Definition 1.1. Let $G = (V, E)$ be a graph and let $0 < \varepsilon < 1$. A *graph sparsifier* of G is a weighted subgraph H of G, on the same vertex set, such that for every $S \subseteq V$,

$$(1 - \varepsilon)\,\mathrm{Cut}_G(S, \bar{S}) \leq \mathrm{Cut}_H(S, \bar{S}) \leq (1 + \varepsilon)\,\mathrm{Cut}_G(S, \bar{S}).$$

A sparsifier is interesting when it has far fewer edges than G. For each fixed ε, sparsifiers with $O(n \log n)$ edges exist; we construct them in Lecture 8. Whether $O(n)$ edges can suffice is one of the questions we take up there.

1.3 Topics we hope to cover

1) The graph as a network of resistors.
2) Random walks and resistance.
3) Matrix theory 101.
    a) Eigenvalues and eigenvectors: the spectral theorem, the Perron–Frobenius theorem and the Courant–Fischer theorem.
4) Graph cuts and eigenvalues.
    a) Cheeger's theorem.
5) Estimating eigenvalues.
    a) Path embedding.
    b) Hardy's inequality.
6) Differential equations and matrix exponentials.
7) Matrix theory 102.
    a) Singular values and singular vectors; Weyl's majorant theorem.
    b) Matrix Chernoff bounds; the Golden–Thompson inequality.
8) Graph sparsifiers.
    a) Sampling by effective resistance.
9) Numerical methods.
    a) The basic iterative method.
    b) The conjugate gradient method.
    c) Preconditioned methods.
    d) Fast Laplacian solvers.

10) Understanding the fundamental eigenvector.
    a) Fiedler's theorem.
11) Random walks and random spanning trees.
    a) The Markov chain tree theorem.
12) Approximate maximum cut using spectral methods.
13) Random walks with resets.
14) Directed Laplacians.
    a) Counting trees.
    b) Making them symmetric.
15) Low-stretch spanning trees.

2 Graph Laplacian, Effective Resistance, Energy

Lecture of January 15, 2020. Based on Gary L. Miller's handwritten notes.

This lecture begins the study of a graph as a network of resistors. We motivate the model with a recommendation problem, recall the laws of electrical circuits, and define the effective resistance between two vertices. Computing effective resistances leads to the Laplacian L of the graph, to harmonic functions and the maximum principle, and to the factorization $L = BCB^\top$ through the boundary operator B. We end by showing that the Laplacian of a connected graph has rank $n-1$ and that its kernel is spanned by the all-ones vector; this guarantees that the linear systems we use to compute effective resistances have solutions.

2.1 Yet another way to search a graph

Depth-first search and breadth-first search are two systematic ways to explore a graph. A third way is a random walk, which at each step moves to a randomly chosen neighbor of the current vertex. We will show that random walks are intimately related to viewing the graph as a network of resistors. This connection lets us apply more mathematics to graphs:

1) linear algebra: linear systems, eigenvalues and eigenvectors, and numerical analysis;

2) matrix Chernoff bounds;

3) the physics of current and energy.

2.2 Resistive model of a graph and random walks

As motivation, consider making recommendations, as Netflix does. The data form a bipartite graph: viewers on one side, movies on the other, and an edge between a viewer and each movie the viewer has ranked, labeled by that ranking (Fig. 1). A higher ranking means a stronger preference. To decide whether to recommend a movie M to a viewer V, we want a number $\mathrm{Score}(V, M)$ that is large when the recommendation is good.

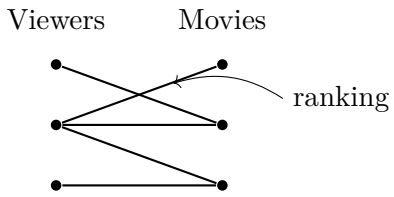


Figure 1: The recommendation graph. Each edge carries the ranking the viewer gave the movie.

We consider five candidate scores. For a path P from V to M, write $P\colon V \to M$.

1) *Inverse distance.* Give each edge the length $w_{ij} = 1/\mathrm{rank}_{ij}$, so that well-ranked edges are short, and set
$$\mathrm{Score}(V, M) = \frac{1}{\mathrm{dist}(V, M)} = \frac{1}{\min_{P\colon V \to M} w(P)},$$
where $w(P)$ is the total length of P.

2) *Bottleneck.* For a path P, let $W(P) = \min_{e \in P} \text{rank}(e)$ be the smallest ranking on P, and set

$$\text{Score}(V, M) = \max_{P:\, V \to M} W(P).$$

3) *Maximum flow.* Let $\text{Score}(V, M)$ be the value of a maximum flow from V to M, with capacities $\text{rank}(e)$.

4) *Effective conductance.* View each edge as a conductor of conductance $\text{rank}(e)$, and let $\text{Score}(V, M)$ be the effective conductance between V and M (Definition 2.2).

5) *Commute time.* Consider a random walk that leaves each vertex along an edge chosen with probability proportional to its rank (Lecture 4). Let $\text{hit}(V, M)$ be the expected number of steps the walk takes to reach M from V, and set

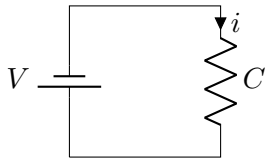
$$\text{Score}(V, M) = \frac{1}{\text{hit}(V, M) + \text{hit}(M, V)}.$$

The first two scores depend only on a single best path, so extra paths from V to M need not improve the score: an extra path that is no better than the best existing one leaves it unchanged. The maximum flow rewards extra paths but ignores their length, so shorter paths do not improve the score. Scores 4 and 5 are equal up to scaling: we show in Lecture 4 that the commute time $\text{hit}(V, M) + \text{hit}(M, V)$ equals the effective resistance between V and M times the total weighted degree of the graph, a constant that does not depend on V or M. Hence score 5 is score 4 divided by this constant. Our hypothesis is that the effective conductance, or equivalently the reciprocal of the commute time, gives a better score than the first three. To test it, we need to give formal definitions, develop the basic theory, give efficient algorithms, and find applications. This lecture starts with the definitions.

2.3 Resistance theory

A resistor has a *resistance* $R > 0$ and a *conductance* $C = 1/R$. If a voltage V is applied across it, a current i flows through it, given by *Ohm's law*:

$$i = CV = \frac{V}{R}.$$



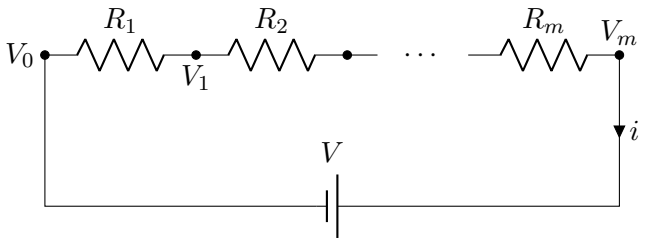
Several resistors can act together like a single one.

Homework 2.1. Show the following two facts.

1) *Resistors in series.* Resistors $R_1, \ldots, R_m$ connected in a chain behave like one resistor of resistance

$$R = R_1 + \cdots + R_m, \qquad \text{that is, of conductance} \qquad C = \frac{1}{1/C_1 + \cdots + 1/C_m},$$

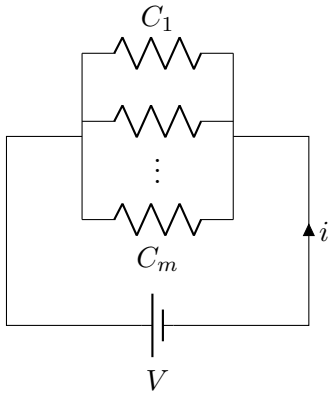
where $C_k = 1/R_k$: a voltage V across the chain drives the current $i = V/R$.



2) *Conductors in parallel.* Conductors $C_1, \dots, C_m$ connected between the same two points behave like one conductor of conductance

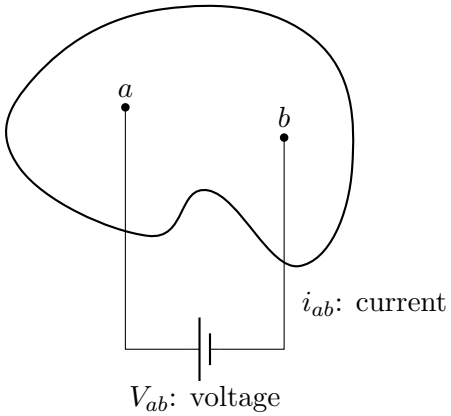
$$C = C_1 + \cdots + C_m :$$

a voltage V drives the current $i = CV$.



2.4 Effective resistance and conductance

Now let G be a whole network of resistors, and let a and b be two of its nodes. Connect a battery between a and b. It holds the voltage difference V_{ab} between a and b, and a current i_{ab} flows into the network at one of them and out at the other.



Definition 2.2. The *effective resistance* and the *effective conductance* between a and b are

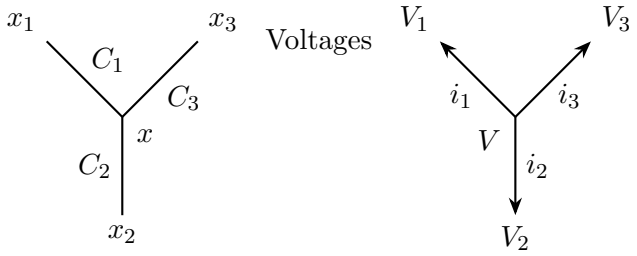
$$R_{ab} \coloneqq \frac{V_{ab}}{i_{ab}}, \qquad C_{ab} \coloneqq \frac{1}{R_{ab}}.$$

Between a and b, the network thus behaves like a single resistor of resistance R_{ab}. The next subsection computes R_{ab} and shows that the ratio V_{ab}/i_{ab} does not depend on the applied voltage.

2.5 Computing effective resistance from a to b

The currents in the network obey *Kirchhoff's current law*: at every node other than a and b, the current flowing in equals the current flowing out.

Example 2.3. Let x be a node joined to three nodes x_1, x_2, x_3 by conductances C_1, C_2, C_3, and let V, V_1, V_2, V_3 be the voltages at x, x_1, x_2, x_3.



By Ohm's law, the current from x to x_k is $i_k = C_k(V - V_k)$ for $k = 1, 2, 3$. The *residual current* at x, the net current flowing out of x, is therefore

$$i_1 + i_2 + i_3 = CV - C_1V_1 - C_2V_2 - C_3V_3, \qquad \text{where } C = C_1 + C_2 + C_3.$$

If x is neither a nor b, Kirchhoff's law says that the residual current is 0, so $CV = C_1V_1 + C_2V_2 + C_3V_3$, that is,

$$V = \frac{C_1}{C}V_1 + \frac{C_2}{C}V_2 + \frac{C_3}{C}V_3.$$

Thus the voltage at x is a convex combination of the voltages at its neighbors, with weights proportional to the conductances.

The general case. Throughout the rest of this lecture, G is an undirected graph without loops or multiple edges, on the vertices $v_1, \ldots, v_n$ with $n \geq 2$, with edge set E and with a conductance $c_{ij} = c_{ji} > 0$ on each edge $(i, j) \in E$. The *weighted degree* of v_i is

$$d(v_i) \coloneqq \sum_{j\colon (i,j)\in E} c_{ij}.$$

Definition 2.4. The *adjacency matrix* A, the *degree matrix* D and the *Laplacian* $L = L(G)$ of G are the $n \times n$ matrices

$$A_{ij} \coloneqq \begin{cases} c_{ij} & \text{if } (i,j) \in E, \\ 0 & \text{otherwise,} \end{cases} \qquad D \coloneqq \begin{pmatrix} d(v_1) & & 0 \\ & \ddots & \\ 0 & & d(v_n) \end{pmatrix}, \qquad L \coloneqq D - A.$$

Equivalently,

$$L_{ij} = \begin{cases} d(v_i) & \text{if } i = j, \\ -c_{ij} & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Let $V \in \mathbb{R}^n$ be a setting of voltages, with voltage V_i at v_i. Then

$$(LV)_i = d(v_i)V_i - \sum_{j\colon (i,j)\in E} c_{ij}V_j = \sum_{j\colon (i,j)\in E} c_{ij}(V_i - V_j),$$

so $(LV)_i$ is the residual current at v_i, exactly as in Example 2.3. Conversely, we may inject currents at the vertices and ask which voltages they produce; this means solving $LV = y$ for the vector y of injected currents. The net injected current must be zero, since every column of L sums to 0 and hence the entries of LV sum to 0.

Our goal is to compute an effective resistance. Numbering the vertices so that the two nodes are v_1 and v_n, we want R_{1n}.

Method 1. Hold v_1 at voltage 1 and v_n at voltage 0, let the other voltages adjust, and measure the current i that flows. In matrix form, we solve

$$L\begin{pmatrix} 1 \\ V_2 \\ \vdots \\ V_{n-1} \\ 0 \end{pmatrix} = \begin{pmatrix} i \\ 0 \\ \vdots \\ 0 \\ -i \end{pmatrix} \qquad (*)$$

for the unknowns $V_2, \ldots, V_{n-1}$ and i, and return $1/i$. Rows 2 to $n-1$ of $(*)$ are Kirchhoff's law at $v_2, \ldots, v_{n-1}$; rows 1 and n say that the current i enters at v_1 and leaves at v_n. Since $i = V/R$ and $V = 1$, the effective resistance is $R_{1n} = 1/i$. Applying a voltage V_{1n} instead of 1 multiplies the solution, and hence the current, by V_{1n}, so the ratio in Definition 2.2 does not depend on V_{1n}. That $(*)$ has exactly one solution, and that $i > 0$, is proved in Proposition 2.14.

We call $(*)$ a *boundary value problem*: the voltages are prescribed at v_1 and v_n, which form the boundary, and the equations at the other vertices determine the rest.

Definition 2.5. Let S be a set of vertices of G. The vertices in S form the *boundary*, and the other vertices are *interior*. A vector $f \in \mathbb{R}^n$, viewed as a function with value f_i at v_i, is *harmonic* with respect to S if $(Lf)_i = 0$ for every interior vertex v_i, that is,

$$d(v_i)\, f_i = \sum_{j\colon (i,j)\in E} c_{ij}\, f_j.$$

When $d(v_i) > 0$, this condition says that f_i is a convex combination of the values of f at the neighbors of v_i, with weights $c_{ij}/d(v_i)$. Rows $2, \ldots, n-1$ of $(*)$ say exactly that its solution is harmonic with respect to $S = \{v_1, v_n\}$.

Theorem 2.6 (Maximum principle)**.** *Let S be a set of vertices of G, and assume that G is connected and $S \neq \emptyset$. If f is harmonic with respect to S, then f attains its maximum and its minimum over all vertices at vertices of S.*

Proof. Since $-f$ is also harmonic, it suffices to treat the maximum. Because G is connected and $n \geq 2$, every vertex has positive degree, so at each interior vertex v_i the value f_i is a convex combination of the values at the neighbors of v_i. Hence v_i has neighbors v_j and v_k with $f_j \leq f_i \leq f_k$. We need slightly more. Let $f_{\max} = \max_{1\leq j\leq n} f_j$ and let $T = \{v_j : f_j = f_{\max}\}$. If an interior vertex lies in T, then so do all of its neighbors, because a convex combination with

positive weights of numbers at most $f_{\max}$ equals $f_{\max}$ only if each of the numbers equals $f_{\max}$. Suppose that T contains no boundary vertex. Then T contains every neighbor of each of its vertices, so T contains all vertices because G is connected, contradicting $S \neq \emptyset$. Hence $f_j = f_{\max}$ for some $v_j \in S$. □

Corollary 2.7 (Uniqueness principle)**.** *Under the hypotheses of Theorem 2.6, if f and g are harmonic with respect to S and agree on S, then $f = g$.*

Proof. The condition in Definition 2.5 is linear in f, so $h = f - g$ is harmonic with respect to S, and $h = 0$ on S. By Theorem 2.6, the maximum and the minimum of h are both attained on S, so both are 0. Hence $h = 0$, that is, $f = g$. □

Method 2. Instead of prescribing voltages, inject one unit of current at v_1 and remove it at v_n. That is, solve

$$LV = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix}$$

and return $R_{1n} = V_1 - V_n$. When G is connected, a solution V exists, the difference $V_1 - V_n$ is the same for every solution, and the two methods return the same value; this is Proposition 2.14 below.

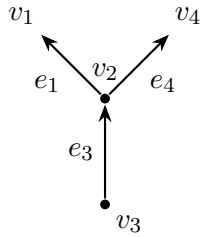
2.6 Another way to view the Laplacian

Fix an orientation of each edge of G, and number the edges $e_1, \ldots, e_m$. For an oriented edge e from u to w, we call u its *tail* and w its *head*.

Definition 2.8 (Boundary operator)**.** The *boundary operator* (or vertex-edge matrix) of G is the $n \times m$ matrix B whose column for the edge e_k has the entry 1 in the row of the head of e_k, the entry -1 in the row of its tail, and 0 elsewhere:

$$B_{n\times m} = \begin{array}{c} \\ v_1 \\ \\ v_n \end{array} \begin{array}{c} \begin{array}{ccc} e_1 & e_k & e_m \end{array} \\ \begin{pmatrix} & -1 & \\ & 1 & \end{pmatrix} \end{array}$$

Example 2.9. For the graph below, with the edges e_1, e_3, e_4 oriented as drawn,



$$B = \begin{array}{c|ccc} & e_1 & e_3 & e_4 \\ \hline v_1 & 1 & 0 & 0 \\ v_2 & -1 & 1 & -1 \\ v_3 & 0 & -1 & 0 \\ v_4 & 0 & 0 & 1 \end{array}$$

Let $c_1, \dots, c_m$ be the conductances of $e_1, \dots, e_m$, and let

$$C \coloneqq \begin{pmatrix} c_1 & & 0 \\ & \ddots & \\ 0 & & c_m \end{pmatrix}.$$

Lemma 2.10. *$L = BCB^\top$, whatever orientation is chosen for each edge.*

Proof. Let $V \in \mathbb{R}^n$ be a setting of voltages. The three matrices act as follows.

1) For each edge e from u to w, $(B^\top V)_e = V_w - V_u$: the vector $B^\top V$ lists the voltage drop across each edge.

2) Hence $(CB^\top V)_e = c_e(V_w - V_u)$: by Ohm's law, $CB^\top V$ lists the current along each edge, counted positive from head to tail.

3) For any $f \in \mathbb{R}^m$ and any vertex v, $(Bf)_v = \sum_{e \text{ with head } v} f_e - \sum_{e \text{ with tail } v} f_e$.

Combining the three, for every vertex v,

$$(BCB^\top V)_v = \sum_{\substack{e \text{ from } u \\ \text{to } v}} c_e(V_v - V_u) + \sum_{\substack{e \text{ from } v \\ \text{to } w}} c_e(V_v - V_w) = \sum_{u:\,(u,v)\in E} c_{uv}(V_v - V_u) = (LV)_v,$$

so $BCB^\top V$ is the residual current at each vertex. Since this holds for every V, we get $L = BCB^\top$. □

From now on, assume that G is connected. We determine the rank and the kernel of L_G, starting from the quadratic form of L.

Lemma 2.11. *For every $x \in \mathbb{R}^n$,*

$$x^\top L x = x^\top BCB^\top x = (B^\top x)^\top C (B^\top x) = \sum_{(i,j)\in E} c_{ij}(x_i - x_j)^2,$$

where the sum has one term for each edge.

Proof. The first equality is Lemma 2.10, and the last holds because C is diagonal and $(B^\top x)_e = \pm(x_i - x_j)$ for the edge e between v_i and v_j. □

Lemma 2.12. *For every $x \in \mathbb{R}^n$, $Lx = 0$ if and only if $x^\top L x = 0$.*

Proof. If $Lx = 0$, then $x^\top L x = 0$. Conversely, suppose that $x^\top L x = 0$. Since C is diagonal with positive entries, it has a square root $C^{1/2}$, and

$$0 = x^\top L x = x^\top BCB^\top x = \left(C^{1/2}B^\top x\right)^\top \left(C^{1/2}B^\top x\right) = \left\|C^{1/2}B^\top x\right\|_2^2.$$

Hence $C^{1/2}B^\top x = 0$, and multiplying on the left by $BC^{1/2}$ gives $BCB^\top x = Lx = 0$. □

Theorem 2.13. *If G is connected, then*

$$\mathrm{Ker}(L_G) = \mathrm{span}\{\mathbf{1}\}, \qquad \mathrm{rank}(L_G) = n - 1,$$

where $\mathbf{1} = (1, \dots, 1)^\top$ is the all-ones vector.

Proof. By Lemmas 2.11 and 2.12, $Lx = 0$ if and only if $\sum_{(i,j)\in E} c_{ij}(x_i - x_j)^2 = 0$. Since every c_{ij} is positive, this holds if and only if $x_i = x_j$ for every edge $(i, j) \in E$. Since G is connected, this holds if and only if all the entries of x are equal, that is, $x \in \operatorname{span}\{\mathbf{1}\}$. Thus $\operatorname{Ker}(L_G) = \operatorname{span}\{\mathbf{1}\}$, and $\operatorname{rank}(L_G) = n - 1$ by the rank-nullity theorem. □

We can now justify both methods for computing R_{1n}.

Proposition 2.14. *Let G be connected.*

1) *The system $LV = (1, 0, \ldots, 0, -1)^\top$ has a solution. Any two solutions differ by a multiple of $\mathbf{1}$, and $V_1 - V_n > 0$ is the same for all of them.*

2) *The system $(*)$ has exactly one solution $(V_2, \ldots, V_{n-1}, i)$, and $i = 1/(V_1 - V_n)$ for every V as in part 1.*

In particular, Methods 1 and 2 both return $R_{1n} = V_1 - V_n > 0$.

Proof. Write $y = (1, 0, \ldots, 0, -1)^\top$. Since L is symmetric, its range is the orthogonal complement of its kernel, which is $\operatorname{span}\{\mathbf{1}\}^\perp$ by Theorem 2.13. The entries of y sum to 0, so $y \perp \mathbf{1}$, and $LV = y$ has a solution. Two solutions differ by an element of $\operatorname{Ker}(L) = \operatorname{span}\{\mathbf{1}\}$, which does not change $V_1 - V_n$. For a solution V,

$$V_1 - V_n = V^\top y = V^\top LV \geq 0$$

by Lemma 2.11, and $V^\top LV \neq 0$ by Lemma 2.12, because $LV = y \neq 0$. This proves part 1.

For part 2, let V be as in part 1 and set $U = (V - V_n\mathbf{1})/(V_1 - V_n)$. Then $U_1 = 1$, $U_n = 0$, and $LU = y/(V_1 - V_n)$, so $(U_2, \ldots, U_{n-1})$ together with $i = 1/(V_1 - V_n)$ solves $(*)$. If (U', i') is another solution, then $U - U'$ is harmonic with respect to $S = \{v_1, v_n\}$ and vanishes on S, so $U' = U$ by Corollary 2.7, and $i' = (LU')_1 = (LU)_1 = i$. □

3 Energy and Rayleigh's Monotonicity Law

Lecture of January 17, 2020. Based on Gary L. Miller's handwritten notes.

This lecture continues the electrical view of a graph from Lecture 2. We define the energy dissipated in a network and split the space of all flows into potential flows and circulations, which are orthogonal with respect to the resistances. From this splitting we prove Thomson's principle: among all unit flows between two vertices, the potential flow, which is the electrical current, has the least energy, and this energy is the effective resistance. Rayleigh's monotonicity law follows: increasing the resistances cannot decrease an effective resistance.

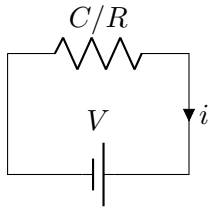
Throughout the lecture, G is a connected graph on the vertices $v_1, \ldots, v_n$, with $n \geq 2$, whose edge set E consists of m edges $e_1, \ldots, e_m$, each with a fixed orientation. Edge e_k has conductance $c_k > 0$ and resistance $R_k = 1/c_k$. As in Lecture 2, B is the $n \times m$ boundary operator, $L = BCB^\top$ is the Laplacian, and

$$C = \begin{pmatrix} c_1 & & 0 \\ & \ddots & \\ 0 & & c_m \end{pmatrix}, \qquad R = \begin{pmatrix} R_1 & & 0 \\ & \ddots & \\ 0 & & R_m \end{pmatrix}, \qquad RC = I.$$

For a vertex x, $\chi_x \in \mathbb{R}^n$ is the standard basis vector with entry 1 at x.

3.1 Current and energy (power dissipation)

In mechanics, the rate at which energy is spent (the power) is force times speed. In a circuit, the corresponding product is voltage times current. Consider a single resistor of conductance C and resistance $R = 1/C$, with voltage V across it and current i through it:



The energy it dissipates per unit time is

$$\begin{aligned} \mathcal{E} &= V \cdot i \\ &= CV^2 && \text{since } i = CV, \\ &= i^2 R && \text{since } V = iR. \end{aligned}$$

As is customary in this subject, we call $\mathcal{E}$ simply the *energy*.

Now consider a network with voltages $V \in \mathbb{R}^n$. The current from x to y along an edge is $i_{xy} = c_{xy}(V_x - V_y)$, and the energy of the network is the sum of the energies of its resistors:

$$\mathcal{E} = \frac{1}{2} \sum_{x,y} |i_{xy}| \cdot |V_x - V_y|,$$

where the sum runs over ordered pairs of adjacent vertices; each edge appears twice, which the factor $\frac{1}{2}$ corrects. Since $|i_{xy}| \cdot |V_x - V_y| = c_{xy}(V_x - V_y)^2$, the energy is the quadratic form of the Laplacian (Lecture 2):

$$V^\top L V = V^\top B C B^\top V = (B^\top V)^\top C (B^\top V) = \sum_{\text{edges } (x,y)} c_{xy}(V_x - V_y)^2 = \mathcal{E}.$$

In terms of the vector $f = CB^\top V$ of edge currents, $RC = I$ gives

$$f^\top R f = V^\top B C R C B^\top V = V^\top B C B^\top V = V^\top L V,$$

so the energy is also $\sum_k R_k f_k^2$, the sum of $i^2 R$ over the resistors.

3.2 Two types of flow

Definition 3.1. A *flow* on G is a function $f\colon E \to \mathbb{R}$ on the oriented edges, that is, a vector $f \in \mathbb{R}^m$. The value f_e is the amount flowing along e from its head to its tail, the sign convention used for currents in Lecture 2; a negative value means flow in the opposite direction. For a vertex x, the *net flow* out of x is

$$f_x \coloneqq (Bf)_x = \sum_{e \text{ with head } x} f_e - \sum_{e \text{ with tail } x} f_e,$$

the total flow leaving x toward its neighbors.

The set of all flows is the m-dimensional vector space $\mathbb{R}^m$. It contains two important subspaces.

Definition 3.2. The *potential flows* are the flows produced by voltages:

$$P_G \coloneqq \{CB^\top V \mid V \in \mathbb{R}^n\}.$$

Since G is connected, P_G is an $(n-1)$-dimensional space. Indeed, C is invertible, so $\dim P_G = \operatorname{rank}(B^\top)$. Moreover, $B^\top x = 0$ if and only if x has equal entries at the two ends of every edge, which for a connected graph means that $x \in \operatorname{span}\{\mathbf{1}\}$. Hence $\operatorname{rank}(B^\top) = n - 1$.

Definition 3.3. The *circulations* are the flows with net flow 0 at every vertex:

$$C_G \coloneqq \{f \in \mathbb{R}^m \mid Bf = 0\}.$$

Lemma 3.4. *The space C_G of circulations has dimension $m - n + 1$.*

Proof. Fix a spanning tree T of G, and call the edges in $E \setminus T$ the *nontree edges*.

1) C_G is a subspace of $\mathbb{R}^m$, since it is the kernel of B.

2) A spanning tree has $n - 1$ edges, so $|E \setminus T| = m - n + 1$.

3) Every flow on the nontree edges can be extended to a circulation. For a nontree edge e from u to w, the path in T from w to u forms a cycle together with e. Let γ_e be the flow that sends one unit around this cycle, through e in the direction of e: it is $+1$ on the cycle edges oriented along the direction of travel, -1 on the cycle edges oriented against it, and 0 elsewhere. At each vertex of the cycle one unit arrives and one unit leaves, so $B\gamma_e = 0$. Moreover, γ_e is 1 on e and 0 on every other nontree edge. Hence, for given values g_e on the nontree edges, the circulation $\sum_{e \in E \setminus T} g_e \gamma_e$ takes the value g_e on each nontree edge e.

4) If $f, g \in C_G$, then $f \restriction (E \setminus T) = g \restriction (E \setminus T)$ if and only if $f = g$. Indeed, $h = f - g$ is a circulation that vanishes on the nontree edges. Suppose that $h \neq 0$. The edges e with $h_e \neq 0$ all lie in T, so they form a nonempty forest, and a nonempty forest has a vertex x that lies on exactly one of its edges, say e. Then $(Bh)_x = \pm h_e \neq 0$, which contradicts $h \in C_G$.

By items 3 and 4, restricting a circulation to the nontree edges is a linear bijection from C_G onto $\mathbb{R}^{E \setminus T}$. Hence $\dim C_G = |E \setminus T| = m - n + 1$. □

The two subspaces are orthogonal with respect to the inner product given by R.

Claim 3.5. *If $f_c \in C_G$ and $g_p \in P_G$, then $f_c^\top R g_p = 0$.*

Proof. There is a $V \in \mathbb{R}^n$ such that $g_p = CB^\top V$. Since $RC = I$ and $Bf_c = 0$,

$$f_c^\top R g_p = f_c^\top RCB^\top V = f_c^\top B^\top V = (Bf_c)^\top V = 0^\top V = 0.$$ □

Proposition 3.6. *Every flow $f \in \mathbb{R}^m$ can be written in exactly one way as*

$$f = f_c + f_p, \qquad f_c \in C_G,\ f_p \in P_G.$$

Proof. If $f \in C_G \cap P_G$, then $f^\top Rf = 0$ by Claim 3.5, and $f = 0$ because R is diagonal with positive entries. So the sum $C_G + P_G$ is direct, and its dimension is $(m - n + 1) + (n - 1) = m$ by Lemma 3.4 and the dimension of P_G. Hence $C_G \oplus P_G = \mathbb{R}^m$. □

Definition 3.7. Let $a \neq b$ be vertices. A flow f is a *unit flow* from a to b if

1) $f_a = -f_b = 1$, and

2) $f_x = 0$ for every vertex $x \neq a, b$,

that is, if $Bf = \chi_a - \chi_b$: one unit leaves a, one unit arrives at b, and flow is conserved at every other vertex.

Lemma 3.8. *Let $a \neq b$. There is exactly one unit potential flow from a to b, namely $f = CB^\top V$ for any $V \in \mathbb{R}^n$ with $LV = \chi_a - \chi_b$.*

Proof. A potential flow $CB^\top V$ is a unit flow from a to b if and only if $BCB^\top V = LV = \chi_a - \chi_b$. By Proposition 2.14, applied with a and b in the roles of v_1 and v_n, such vectors V exist, and any two of them differ by a multiple of $\mathbf{1}$. Since $B^\top \mathbf{1} = 0$, adding a multiple of $\mathbf{1}$ to V does not change $CB^\top V$. □

3.3 Thomson's principle

Theorem 3.9 (Thomson's principle)**.** *Let $a \neq b$ be vertices, let f be the unit potential flow from a to b, and let g be any unit flow from a to b. Then*

$$f^\top Rf \leq g^\top Rg.$$

Proof. We first show that $g = f + f_c$ for a circulation f_c. By Proposition 3.6, $g = g_p + f_c$ with $g_p \in P_G$ and $f_c \in C_G$. Since $Bf_c = 0$, we have $Bg_p = Bg = \chi_a - \chi_b$, so g_p is a unit potential flow from a to b. By Lemma 3.8, $g_p = f$. Now $f_c^\top Rf = 0$ by Claim 3.5, so

$$\begin{aligned} g^\top Rg &= (f + f_c)^\top R(f + f_c) \\ &= f^\top Rf + 2f_c^\top Rf + f_c^\top Rf_c \\ &= f^\top Rf + f_c^\top Rf_c \\ &\geq f^\top Rf, \end{aligned}$$

where the last step holds because R is diagonal with positive entries. □

In words, among all unit flows from a to b, the potential flow, which is the electrical current, dissipates the least energy. Its energy is the effective resistance.

Definition 3.10. For vertices $a \neq b$, let f_p be the unit potential flow from a to b. The *effective resistance* from a to b is

$$\mathrm{ER}_{ab} \coloneqq f_p^\top Rf_p.$$

This agrees with the effective resistance R_{ab} of Lecture 2, so either description can serve as the definition.

Proposition 3.11. *Let $a \neq b$, and let $V \in \mathbb{R}^n$ satisfy $LV = \chi_a - \chi_b$. Then $\mathrm{ER}_{ab} = \mathrm{ER}_{ba} = V_a - V_b = R_{ab}$.*

Proof. By Lemma 3.8, $f_p = CB^\top V$. As in the computation of the energy above,

$$\mathrm{ER}_{ab} = f_p^\top Rf_p = V^\top LV = V^\top(\chi_a - \chi_b) = V_a - V_b.$$

The vector $-V$ satisfies $L(-V) = \chi_b - \chi_a$, so the same computation with a and b exchanged gives $\mathrm{ER}_{ba} = (-V_b) - (-V_a) = V_a - V_b$. By Method 2 of Lecture 2 (Proposition 2.14, with $a = v_1$ and $b = v_n$), $R_{ab} = V_a - V_b$. □

3.4 Rayleigh's monotonicity law

Let R and $\bar{R}$ be two assignments of positive resistances to the edges of G, and let G_R and $G_{\bar{R}}$ be the resulting networks. They have the same vertices, edges and orientations, and hence the same boundary operator B. Write ER_{ab} and $\overline{\mathrm{ER}}_{ab}$ for the effective resistances in G_R and in $G_{\bar{R}}$, and write $\bar{R} \geq R$ if $\bar{R}_e \geq R_e$ for every edge e.

Theorem 3.12 (Rayleigh's monotonicity law)**.** *If $\bar{R} \geq R$, then $\overline{\mathrm{ER}}_{ab} \geq \mathrm{ER}_{ab}$ for all vertices $a \neq b$.*

Proof. Let f be the unit potential flow from a to b in G_R, and g the one in $G_{\bar{R}}$. Whether a flow is a unit flow from a to b depends only on B, so g is also a unit flow from a to b in G_R. Hence

$$\begin{aligned} \overline{\mathrm{ER}}_{ab} = g^\top \bar{R} g &= \sum_e g_e^2 \bar{R}_e \\ &\geq \sum_e g_e^2 R_e && \text{since } \bar{R} \geq R, \\ &\geq \sum_e f_e^2 R_e && \text{by Thomson's principle in } G_R, \\ &= f^\top Rf = \mathrm{ER}_{ab}\,. \end{aligned}$$

□

Homework 3.13. For vertices $a \neq b$ let R_{ab} be the effective resistance between them, and set $R_{aa} = 0$. Show that $(a, b) \mapsto R_{ab}$ is a metric on the vertices of G, that is, for all vertices a, b, c:

1) $R_{ab} \geq 0$;

2) $R_{ab} = 0$ if and only if $a = b$;

3) $R_{ab} = R_{ba}$;

4) $R_{ac} \leq R_{ab} + R_{bc}$.

4 Random Walks on Graphs and Commute Time

Lecture of January 22, 2020. No Spring 2020 notes are available for this lecture, so it is based on Gary L. Miller's Spring 2021 handwritten notes.

A random walk on a weighted graph moves from vertex to vertex, choosing each step at random with probabilities proportional to the edge weights. This lecture introduces random walks and their basic parameters, and then shows that for symmetric weights the walk is governed by the electrical network of Lectures 2 and 3. The probability of reaching one vertex before another is a voltage, the expected net number of times the walk crosses an edge is a current, and the hitting times to a vertex are the voltages of a single Laplacian system. As a consequence, the commute time between two vertices equals their effective resistance times the total conductance of the graph.

4.1 Random walks on graphs

Let G be a graph, possibly directed, on the vertices $v_1, \dots, v_n$, with edge set E and a weight $w_{ij} > 0$ on each edge $(i,j) \in E$. Assume that every vertex has at least one outgoing edge, and define

$$w_i = w(v_i) \coloneqq \sum_{j\colon (i,j)\in E} w_{ij}, \qquad P_{ij} \coloneqq \frac{w_{ij}}{w_i},$$

with $P_{ij} = 0$ if $(i,j) \notin E$. Then $P_{ij} \geq 0$ and $\sum_j P_{ij} = 1$ for every i.

Definition 4.1. A *random walk* on G is a sequence $X_0, X_1, X_2, \dots$ of random vertices with the following property: whenever the walk is at v_i, it moves next to v_j with probability P_{ij}, independently of its earlier history.

4.2 A small example

As a small example, consider looking for a lost phone in a house with three rooms: a bedroom, a kitchen and a living room. The weighted graph on the left of Fig. 2 has a loop of weight 1 at each room, an edge of weight 1 between the bedroom and the kitchen, and an edge of weight 2 between the kitchen and the living room. The resulting transition probabilities are on the right.

Order the vertices as bedroom, kitchen, living room. The weighted adjacency matrix A (with $A_{ij} = w_{ij}$) and the diagonal matrix D of the weights w_i are

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix}, \qquad D = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 3 \end{pmatrix},$$

and

$$A^\top D^{-1} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{2} & \frac{1}{4} & \frac{2}{3} \\ 0 & \frac{1}{2} & \frac{1}{3} \end{pmatrix}.$$

The entry of $A^\top D^{-1}$ in row j and column i is $A_{ij}/w_i = P_{ij}$, so each column lists the transition probabilities out of one room. The vector $\pi = (\frac{2}{9}, \frac{4}{9}, \frac{3}{9})^\top$ shown in brackets in Fig. 2 is the *stationary distribution* of the walk: it satisfies $A^\top D^{-1}\pi = \pi$, since A is symmetric and so $A^\top D^{-1}\pi = \frac{1}{9}A\mathbf{1} = \frac{1}{9}(2,4,3)^\top = \pi$. The next two subsections explain why this equation means that the distribution π does not change from one step to the next.

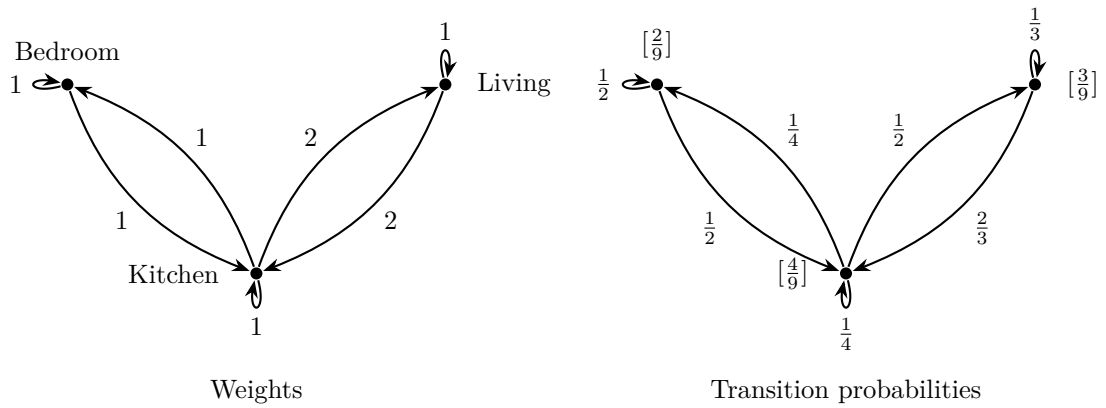


Figure 2: The phone example. Left: the weighted graph. Right: the transition probabilities P_{ij}, with the stationary distribution in brackets.

4.3 Large graphs

Random walks are also useful on graphs far too large to write down. For example, let the vertices be the 52! orderings of a deck of 52 cards, and let P_{ij} be the probability that one shuffle turns the ith ordering into the jth. Professional players start a game after five shuffles. Why so few shuffles should suffice is a question about how fast this walk approaches its stationary distribution, which we study in Lecture 5.

4.4 Two views of a random walk

There are two ways to think of a random walk.

1) The *particle view*, which is Definition 4.1: a single walker moving from vertex to vertex.

2) The *wave view*: a probability distribution over the vertices, or equivalently a large number of independent walkers moving simultaneously.

In the wave view, let $x^{(t)} \in \mathbb{R}^n$ be the distribution at time t, that is, $x_i^{(t)} = \Pr[X_t = v_i]$. Since the walk reaches v_j at time $t+1$ by moving from some v_i at time t,

$$x_j^{(t+1)} = \sum_i P_{ij}\, x_i^{(t)} = \sum_i \frac{A_{ij}}{w_i}\, x_i^{(t)}, \qquad \text{that is,} \qquad x^{(t+1)} = A^\top D^{-1} x^{(t)},$$

where A is the weighted adjacency matrix and D is the diagonal matrix of the weights w_i. A probability vector π with $A^\top D^{-1}\pi = \pi$ is therefore a distribution that stays the same forever once the walk has it; this is a *stationary distribution*.

4.5 Important parameters

For a random walk started at $X_0 = v_i$, let $\tau_j = \min\{t \geq 0 : X_t = v_j\}$ be the first time it visits v_j.

Definition 4.2. Let v_i and v_j be vertices of G.

1) The *access time*, or *hitting time*, from v_i to v_j is $H(i,j) = H_{ij} \coloneqq \mathbb{E}[\tau_j]$ for the walk started at v_i: the expected time to visit v_j starting at v_i. In particular, $H(i,i) = 0$.

2) The *commute time* between v_i and v_j is

$$K(i,j) := H(i,j) + H(j,i).$$

3) The *cover time* is the expected time for the walk to visit every vertex, maximized over all starting vertices.

A fourth parameter, the *mixing rate*, measures how fast the distribution $x^{(t)}$ approaches a stationary distribution; it is the subject of Lecture 5. This lecture computes hitting times and commute times.

4.6 Random walks: the symmetric case

The key idea is to view a random walk as a walk on a network of conductors. From now on, the input is a graph G on $n \geq 2$ vertices, without loops or multiple edges, with a conductance c_{xy} on each edge $\{x, y\}$, such that

1) $c_{xy} = c_{yx} > 0$, so that G is undirected, and

2) G is connected.

We write $c_{xy} = 0$ when x and y are not adjacent. The walk uses the conductances as weights:

$$C_x := \sum_y c_{xy}, \qquad P_{xy} = \frac{c_{xy}}{C_x}.$$

Thus C_x is the weighted degree $d(x)$ of Lecture 2, and the Laplacian L of G has diagonal entries C_x and off-diagonal entries $-c_{xy}$.

All the expected times below are finite, by the following lemma.

Lemma 4.3. *Let S be a nonempty set of vertices of G. For every starting vertex, the walk visits S with probability 1, and the expected time until its first visit to S is finite.*

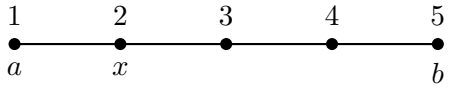
Proof. Let $p > 0$ be the smallest nonzero transition probability P_{xy}. Since G is connected, from every vertex there is a path of at most $n-1$ edges to S, and the walk follows such a path with probability at least $\delta = p^{n-1} > 0$. Hence, whatever happened before, the walk reaches S during each block of $n-1$ consecutive steps with probability at least δ. So the probability that the walk has not reached S by time $k(n-1)$ is at most $(1-\delta)^k$, which tends to 0, and the expected time to reach S is at most $\sum_{k\geq 0}(n-1)(1-\delta)^k = (n-1)/\delta$. □

4.7 The pinball machine problem

Fix two distinct vertices a and b. Think of the walk as a ball in a pinball machine that bounces around the graph until it falls into one of two holes, a or b.

Definition 4.4. For a vertex x, let h_x be the probability that the random walk started at x visits a before b. A visit at time 0 counts, so $h_a = 1$ and $h_b = 0$.

Example 4.5. Let G be the path on the vertices $1, 2, 3, 4, 5$ with unit conductances, and let $a = 1$, $x = 2$ and $b = 5$.



Here $h_a = 1$ and $h_b = 0$. Since x is next to a and three steps away from b, we expect $h_x > \frac{1}{2}$. Example 4.10 shows that $h_x = \frac{3}{4}$.

4.8 A recurrence for h_x

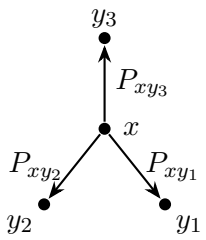
Claim 4.6. *For every vertex $x \notin \{a, b\}$,*

$$h_x = \sum_y P_{xy} h_y.$$

Since $P_{xy} \geq 0$, $\sum_y P_{xy} = 1$, and $P_{xy} > 0$ exactly when y is a neighbor of x, the value h_x is a convex combination of the values h_y at the neighbors y of x.

The idea is to condition on the first step. From x, the walk moves to a neighbor y_i with probability P_{xy_i}, and from y_i it visits a before b with probability h_{y_i}:

$$h_x = \sum_{i=1}^{3} P_{xy_i} h_{y_i}$$



The following proof makes this precise by counting walks.

Proof. A *walk* is a finite sequence of vertices $y_0, y_1, \ldots, y_k$ in which consecutive vertices are adjacent; its probability is $P_{y_0y_1} P_{y_1y_2} \cdots P_{y_{k-1}y_k}$, the probability that the random walk started at y_0 begins with it. For vertices x, s, t, let $W(x, s, t)$ be the set of walks that

1) start at x,

2) end at s, and visit s only at the end, and

3) do not visit t.

No walk in $W(x, a, b) \cup W(x, b, a)$ is a proper initial segment of another, since each of them stops at its first visit to $\{a, b\}$. So the events that the random walk begins with these walks are disjoint, and we write $\Pr[W]$ for the sum of the probabilities of the walks in a set W. The random walk from x visits a before b exactly when it begins with a walk in $W(x, a, b)$, and it visits $\{a, b\}$ at all exactly when it begins with a walk in $W(x, a, b) \cup W(x, b, a)$. So $h_x = \Pr[W(x, a, b)]$. By the next claim, h_x is also the conditional probability

$$\Pr\big[W(x, a, b) \mid W(x, a, b) \cup W(x, b, a)\big] = \frac{\Pr[W(x, a, b)]}{\Pr[W(x, a, b) \cup W(x, b, a)]}$$

of visiting a before b, given that the walk visits $\{a, b\}$.

Claim 4.7. *For all vertices $x, y \notin \{a, b\}$,*

$$\Pr\big[W(x, a, b) \cup W(x, b, a)\big] = \Pr\big[W(y, a, b) \cup W(y, b, a)\big].$$

In fact, $\Pr\big[W(x, a, b) \cup W(x, b, a)\big] = 1$ for every vertex x.

Proof. The probability $\Pr\big[W(x, a, b) \cup W(x, b, a)\big]$ is the probability that the random walk from x ever visits $\{a, b\}$, which is 1 by Lemma 4.3. □

Claim 4.8. *For every vertex $x \notin \{a, b\}$,*

$$\Pr[W(x, a, b)] = \sum_y P_{xy} \Pr[W(y, a, b)].$$

Proof. Since $x \neq a$, every walk in $W(x, a, b)$ has a second vertex y, a neighbor of x, and removing its first vertex leaves a walk in $W(y, a, b)$. Conversely, for each neighbor y of x, putting x in front of a walk in $W(y, a, b)$ gives a walk in $W(x, a, b)$, since $x \notin \{a, b\}$. This bijection multiplies probabilities by P_{xy}, and $P_{xy} = 0$ when y is not a neighbor of x. Here $W(a, a, b)$ consists of the one-vertex walk (a), of probability 1, and $W(b, a, b)$ is empty. □

Since $h_x = \Pr[W(x, a, b)]$ for every vertex x, including $h_a = 1$ and $h_b = 0$, Claim 4.8 is exactly the recurrence $h_x = \sum_y P_{xy} h_y$ for $x \notin \{a, b\}$. □

So h is harmonic with respect to the boundary $\{a, b\}$: multiplying the recurrence by C_x gives $C_x h_x = \sum_y c_{xy} h_y$, which is the condition of Definition 2.5. We now construct an electrical problem that satisfies the same recurrence.

4.9 The electrical version

Hold a at voltage $V_a = 1$ and b at voltage $V_b = 0$, and let the other voltages float. By Kirchhoff's law (Lecture 2), for every $x \neq a, b$,

$$V_x = \sum_y \frac{c_{xy}}{C_x} V_y.$$

But $c_{xy}/C_x = P_{xy}$, so V satisfies the same recurrence as h, with the same boundary values. Hence $h = V$.

Theorem 4.9. *Let $a \neq b$, and let V be the voltages with $V_a = 1$, $V_b = 0$ and V harmonic at every other vertex. Then for every vertex x, V_x is the probability that the random walk started at x visits a before b. Equivalently, if the vertices are numbered so that $a = v_1$ and $b = v_n$, and $(V_2, \ldots, V_{n-1}, i)$ solves*

$$L \begin{pmatrix} 1 \\ V_2 \\ \vdots \\ V_{n-1} \\ 0 \end{pmatrix} = \begin{pmatrix} i \\ 0 \\ \vdots \\ 0 \\ -i \end{pmatrix},$$

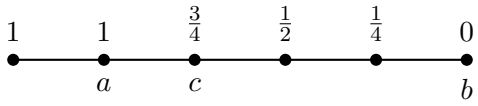
then $h_{v_k} = V_k$ for every k.

Proof. The system has exactly one solution by Proposition 2.14. By Claim 4.6, h and V are both harmonic with respect to $\{a, b\}$, and they agree on $\{a, b\}$. Since G is connected, $h = V$ by the uniqueness principle (Corollary 2.7). □

4.10 Back to the path example

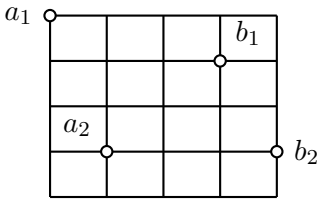
At each vertex other than a and b, the residual current is 0.

Example 4.10. Let G be the path $p_0, p_1, \ldots, p_5$ with unit conductances, and let $a = p_1$, $c = p_2$ and $b = p_5$. The vertex p_0 has only the neighbor a, so $V_{p_0} = V_a = 1$. At p_2, p_3 and p_4, the voltage is the average of the voltages at the two neighbors, so the voltage drops by the same amount across each of the four edges from a to b, namely by $\frac{1}{4}$:



In particular, $V_c = \frac{3}{4}$. A walk from c cannot reach p_0 without visiting a first, so the probability of visiting a before b from c is the same as in Example 4.5, where c plays the role of x. Hence $h_x = \frac{3}{4} > \frac{1}{2}$ there.

In general, a and b may be replaced by several sources and sinks, that is, by two disjoint nonempty sets of vertices, such as $\{a_1, a_2\}$ and $\{b_1, b_2\}$ in the grid below. The probability that the walk visits the first set before the second is the voltage when every vertex of the first set is held at 1 and every vertex of the second at 0; the proof of Theorem 4.9 applies with the boundary $\{a_1, a_2, b_1, b_2\}$.



Thus these probabilities, for all starting vertices at once, are computed by one Laplacian solve.

4.11 Interpretation of current for random walk

Consider one unit of potential current flowing from a to b: let V satisfy $LV = \chi_a - \chi_b$, where χ_x is the standard basis vector of x, and let $i_{xy} = c_{xy}(V_x - V_y)$ be the current from x to y along the edge $\{x, y\}$. This is the unit potential flow from a to b of Lecture 3. The next theorem says what i_{xy} means for the random walk from a to b.

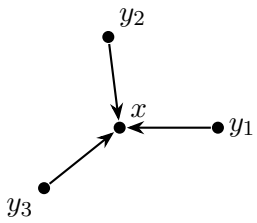
Theorem 4.11. *Let $a \neq b$, and let i be the unit current flow from a to b as above. Consider the random walk started at a and stopped when it first visits b. For every edge $\{x, y\}$, i_{xy} is the expected net number of traversals of the edge from x to y: the expected number of steps from x to y minus the expected number of steps from y to x.*

Proof. Let τ be the first time the walk visits b; then $\mathbb{E}[\tau]$ is finite by Lemma 4.3. For each vertex x, let u_x be the expected number of visits to x before reaching b, that is, of times $t < \tau$ with $X_t = x$; the start counts as a visit to a. Then $u_b = 0$, and each $u_x \leq \mathbb{E}[\tau]$ is finite.

Step 1: a recurrence for u. We leave the following recurrence as homework.

Homework 4.12. Show that $u_x = \sum_y u_y P_{yx}$ for every vertex $x \notin \{a, b\}$.

Note that the column sum $\sum_y P_{yx}$ need not equal 1 (it does when all the weighted degrees C_y are equal), so in general this is not a convex combination.



Step 2: reversibility. Since $c_{xy} = c_{yx}$, for all vertices x and y,

$$C_x P_{xy} = C_x\Big(\frac{c_{xy}}{C_x}\Big) = c_{xy} = c_{yx} = C_y\Big(\frac{c_{yx}}{C_y}\Big) = C_y P_{yx}.$$

Step 3: a voltage. Set $V'_x = u_x/C_x$ for every vertex x. For $x \notin \{a, b\}$, Steps 1 and 2 give

$$u_x = \sum_y u_y P_{yx} = \sum_y \frac{u_y}{C_y} C_y P_{yx} = \sum_y V'_y C_x P_{xy}, \qquad \text{so} \qquad V'_x = \sum_y P_{xy} V'_y.$$

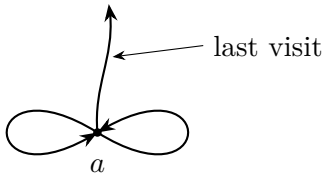
Thus V' is harmonic at every $x \notin \{a, b\}$: the residual current there is 0. Its boundary values are $V'_b = 0$ and $V'_a = u_a/C_a$, for the correct value of u_a.

Step 4: currents count traversals. Let $j_{xy} = c_{xy}(V'_x - V'_y)$ be the current from x to y for the voltages V'. Then

$$j_{xy} = \Big(\frac{u_x}{C_x} - \frac{u_y}{C_y}\Big)c_{xy} = u_x\Big(\frac{c_{xy}}{C_x}\Big) - u_y\Big(\frac{c_{yx}}{C_y}\Big) = u_x P_{xy} - u_y P_{yx}.$$

Each visit to x before time τ is followed by a step to y with probability P_{xy}, so $u_x P_{xy}$ is the expected number of traversals from x to y, and $u_y P_{yx}$ is the expected number from y to x. Hence j_{xy} is the expected net number of traversals from x to y.

Step 5: the net current out of a is 1. We show that $\sum_y j_{ay} = 1$. In every run, the walk starts at a and ends at $b \neq a$. Each visit to a before time τ is followed by a step out of a, and each visit except the one at time 0 is preceded by a step into a; there are no other steps into or out of a. After its last visit to a, the walk leaves a for good:



So every run makes exactly one more step out of a than into a. All these counts are at most τ, so we may take expectations, and Step 4 gives

$$\sum_y j_{ay} = \sum_y u_a P_{ay} - \sum_y u_y P_{ya} = 1.$$

Step 6: conclusion. By Steps 3 and 5, $(LV')_x = \sum_y c_{xy}(V'_x - V'_y) = \sum_y j_{xy}$ is 0 for $x \notin \{a, b\}$ and 1 for $x = a$. The entries of LV' sum to 0, so $(LV')_b = -1$. Hence $LV' = \chi_a - \chi_b = LV$, so V' and V differ by a multiple of $\mathbf{1}$ (Theorem 2.13), and $j_{xy} = i_{xy}$ for every edge. By Step 4, this proves the theorem. □

The proof also computes u_a, the expected number of visits to a before reaching b.

Corollary 4.13. *For $a \neq b$, the walk started at a visits a on average $u_a = C_a \cdot \mathrm{ER}_{ab}$ times before reaching b, counting the visit at time 0.*

Proof. Number the vertices so that $a = v_1$ and $b = v_n$. Solve

$$LV = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix},$$

and set $V' = V - V_n\mathbf{1}$, so that $V'_n = 0$ and $LV' = LV$. By Step 6 of the proof of Theorem 4.11, the vector $(u_x/C_x)_x$ solves the same system and vanishes at b. Two solutions differ by a multiple of $\mathbf{1}$, which must be 0 here, so $V'_1 = u_1/C_1$, that is, $u_1 = V'_1 C_1$. Finally, $V'_1 = V_1 - V_n = \mathrm{ER}_{ab}$ by Propositions 2.14 and 3.11. Hence $u_a = C_a \cdot \mathrm{ER}_{ab}$. □

4.12 How to compute hitting time

Fix a vertex b, and write $H_x = H(x, b)$ for the hitting time from x to b (Definition 4.2); it is finite by Lemma 4.3. Conditioning on the first step gives a recurrence:

$$\begin{aligned} H_b &= 0, \\ H_x &= 1 + \sum_y P_{xy} H_y \qquad (x \neq b), \end{aligned}$$

because the walk takes one step, to y with probability P_{xy}, and then needs H_y more steps on average.

Think of H_x as a voltage V_x. Then $V_b = 0$ and, for $x \neq b$,

$$V_x = 1 + \sum_y \frac{c_{xy}}{C_x} V_y, \qquad \text{that is,} \qquad \underbrace{C_x V_x - \sum_y c_{xy} V_y}_{(LV)_x} = \underbrace{C_x}_{\text{residual current}}.$$

These are $n-1$ constraints. Number the vertices so that $b = v_n$, and let $C = \sum_x C_x$ be the total conductance. The entries of LV sum to 0, so the remaining row is $(LV)_n = -\sum_{x \neq v_n} C_x = C_n - C$. Hence $V = H$ satisfies

$$LV = \begin{pmatrix} C_1 \\ \vdots \\ C_{n-1} \\ C_n - C \end{pmatrix}, \qquad V_n = 0.$$

Algorithm 4.14 (Hitting times to v_n)**.** Input: a connected graph G with symmetric conductances. Output: $H(x, v_n)$ for every vertex x.

1) Solve

$$LV = \begin{pmatrix} C_1 \\ \vdots \\ C_{n-1} \\ C_n - C \end{pmatrix}.$$

2) Replace V by $V - V_n\mathbf{1}$, so that $V_n = 0$.

3) Return V_x as $H(x, v_n)$, for every vertex x.

The algorithm is correct. The right-hand side sums to 0, so it is orthogonal to $\mathrm{Ker}(L) = \mathrm{span}\{\mathbf{1}\}$; since L is symmetric, the system has a solution, and any two solutions differ by a multiple of $\mathbf{1}$ (Theorem 2.13). The vector of hitting times is the solution with $V_n = 0$.

4.13 Commute time

Theorem 4.15. *Let G be a connected graph with symmetric conductances, let $a \neq b$, let $C = \sum_x C_x = 2\sum_{\text{edges } e} c_e$ be its total conductance, and let ER_{ab} be the effective resistance between a and b (Lecture 3). Then the commute time between a and b is*

$$K(a, b) = C \cdot \mathrm{ER}_{ab}\,.$$

If every edge has conductance 1, then $C = 2m$, and so $K(a, b) = 2m\ \mathrm{ER}_{ab}$.

Proof. The total conductance is $2\sum_e c_e$ because each edge contributes its conductance to C_x at both of its ends. Number the vertices so that $a = v_1$ and $b = v_n$.

Solution 1. Solve

$$LV^b = \begin{pmatrix} C_1 \\ C_2 \\ \vdots \\ C_n - C \end{pmatrix}, \qquad LV^a = \begin{pmatrix} C_1 - C \\ C_2 \\ \vdots \\ C_n \end{pmatrix}.$$

By Algorithm 4.14, and by the same algorithm with the roles of v_1 and v_n exchanged,

$$H(1, n) = V_1^b - V_n^b \qquad \text{and} \qquad H(n, 1) = V_n^a - V_1^a;$$

these differences do not depend on which solutions we take. Setting $V = V^b - V^a$, the commute time is

$$K(1, n) = H(1, n) + H(n, 1) = (V^b - V^a)_1 - (V^b - V^a)_n = V_1 - V_n.$$

Solution 2. Only $V^b - V^a$ matters, and

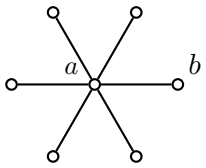
$$L(V^b - V^a) = LV^b - LV^a = \begin{pmatrix} C_1 \\ \vdots \\ C_n - C \end{pmatrix} - \begin{pmatrix} C_1 - C \\ \vdots \\ C_n \end{pmatrix} = \begin{pmatrix} C \\ 0 \\ \vdots \\ 0 \\ -C \end{pmatrix} = C \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix}.$$

So it suffices to solve

$$LV = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix}$$

and return $C(V_1 - V_n)$, since CV differs from $V^b - V^a$ by a multiple of $\mathbf{1}$. But $V_1 - V_n = \mathrm{ER}_{1n}$ by Propositions 2.14 and 3.11. Hence $K(a, b) = C \cdot \mathrm{ER}_{ab}$. □

Example 4.16. Let G be the star with center a and $n-1$ leaves, with unit conductances, and let b be a leaf.



A unit current from a to b flows entirely through the edge ab: each other leaf lies on a single edge and has net current 0, so that edge carries no current. Hence $\mathrm{ER}_{ab} = 1$. There are $m = n-1$ edges, so Theorem 4.15 gives

$$K(a,b) = 2(n-1).$$

4.14 Another example

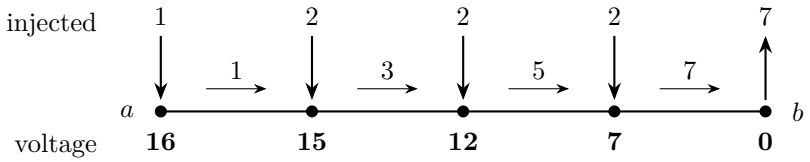


Figure 3: Hitting times to b on the path with unit conductances. Above the vertices: the injected currents (7 units leave at b). Above the edges: the edge currents. Below the vertices, in bold: the voltages, which are the hitting times to b.

Example 4.17. Let G be the path $a = p_0, p_1, p_2, p_3, p_4 = b$ with unit conductances, so $m = 4$ and $C = 8$. To compute the hitting times to b, Algorithm 4.14 injects the currents $C_x = 1, 2, 2, 2$ at $p_0, \ldots, p_3$ and the current $C_b - C = 1 - 8 = -7$ at b; that is, 7 units leave at b (Fig. 3). By Kirchhoff's law, the current on the edge from p_k to p_{k+1} is the total current injected at $p_0, \ldots, p_k$, which gives $1, 3, 5, 7$. With unit conductances, the voltage drop across each edge equals its current, so with $V_b = 0$ the voltages at p_3, p_2, p_1, p_0 are

$$7, \qquad 7+5 = 12, \qquad 12+3 = 15, \qquad 15+1 = 16.$$

These voltages are the hitting times to b; in particular $H(a,b) = 16$. Reflecting the path exchanges a and b, so $H(b,a) = 16$ as well, and the commute time is $K(a,b) = 32$. This agrees with Theorem 4.15: four unit resistors in series have $\mathrm{ER}_{ab} = 4$ (Homework 2.1), and

$$K(a,b) = 2m\ \mathrm{ER}_{ab} = 2 \cdot 4 \cdot 4 = 32.$$

5 Random Walks and Mixing Times

Lecture of January 24, 2020. Based on Gary L. Miller's handwritten notes.

A random walk on a weighted graph moves from its current vertex to a neighbor chosen with probability proportional to the weight of the connecting edge. This lecture writes one step of the walk as multiplication by the transition matrix AD^{-1}, exhibits a stationary distribution π, and asks when and how fast the walk converges to π. A change of variables makes the transition matrix symmetric, so that the spectral theorem applies: the error then shrinks by a factor λ per step, where λ is the largest absolute value of an eigenvalue other than the top one. Along the way we state the Perron–Frobenius theorem, whose symmetric version is proved in Lecture 6, compare the error in the ℓ_1 and ℓ_2 norms, prove the spectral theorem, and discuss the walk on a path.

5.1 Random walk as a matrix-vector product

Throughout this lecture, $G = (V, E, w)$ is an undirected graph with vertex set $V = \{v_1, \dots, v_n\}$ and positive edge weights. Its adjacency matrix A has $A_{ij} = w_{ij}$ if $(i, j) \in E$ and $A_{ij} = 0$ otherwise, and $D = \operatorname{diag}(d_1, \dots, d_n)$ with $d_i = \sum_j w_{ij}$. We assume that every vertex has positive degree, so that D is invertible.

The *random walk* on G moves from v_i to v_j with probability w_{ij}/d_i. For $t \geq 0$, let $p_i^{(t)}$ be the probability that the walk is at v_i at time t, and let $p^{(t)} = (p_1^{(t)}, \dots, p_n^{(t)})^\top$. The *starting distribution* $p^{(0)}$ is any vector with $p^{(0)} \geq 0$ and $\sum_i p_i^{(0)} = 1$.

Claim 5.1. *For every $t \geq 0$, $p^{(t+1)} = A^\top D^{-1} p^{(t)}$.*

Proof. Multiplying $A^\top$ on the right by D^{-1} divides column i of $A^\top$ by d_i:

$$\underbrace{\begin{pmatrix} | & & | & & | \\ w_1 & \cdots & w_i & \cdots & w_n \\ | & & | & & | \end{pmatrix}}_{A^\top} \underbrace{\begin{pmatrix} d_1^{-1} & & 0 \\ & \ddots & \\ 0 & & d_n^{-1} \end{pmatrix}}_{D^{-1}}, \qquad \text{column } i \text{ of the product} = \begin{pmatrix} w_{i1}/d_i \\ \vdots \\ w_{in}/d_i \end{pmatrix},$$

where w_i denotes column i of $A^\top$. Hence

$$\Pr[\text{step from } v_i \text{ to } v_j] = \frac{w_{ij}}{d_i} = (A^\top D^{-1})_{ji}.$$

By the law of total probability, $p_j^{(t+1)} = \sum_i p_i^{(t)} w_{ij}/d_i = \sum_i (A^\top D^{-1})_{ji} p_i^{(t)}$ for every j. □

Column i of $A^\top D^{-1}$ lists the probabilities of the steps out of v_i, so every column sum of $A^\top D^{-1}$ equals $\sum_j w_{ij}/d_i = 1$.

Definition 5.2. The matrix $A^\top D^{-1}$ is the *transition matrix* of the random walk on G.

Since G is undirected, $A = A^\top$, and we also write the transition matrix as AD^{-1}.

5.2 Stationary distributions and convergence

Two questions about the walk are natural. Both concern the distributions that a step of the walk leaves unchanged.

Definition 5.3. A vector $\bar{p} \in \mathbb{R}^n$ with $\bar{p} \geq 0$ and $\sum_i \bar{p}_i = 1$ is a *stationary distribution* of the walk if $AD^{-1}\bar{p} = \bar{p}$.

The two questions are:

1) whether a stationary distribution $\bar{p}$ exists, and

2) whether $\lim_{k\to\infty}(AD^{-1})^k p^{(0)} = \bar{p}$ for every starting distribution $p^{(0)}$.

The answer to the first question is yes. Let $d = \sum_i d_i$ and

$$\pi \coloneqq \begin{pmatrix} d_1/d \\ \vdots \\ d_n/d \end{pmatrix} = \frac{1}{d} D\mathbf{1}.$$

Proposition 5.4. *The vector π is a stationary distribution of the random walk on G.*

Proof. Clearly $\pi \geq 0$ and $\sum_i \pi_i = 1$. Since $(A\mathbf{1})_i = \sum_j w_{ij} = d_i$,

$$A^\top D^{-1}\pi = A^\top D^{-1}\Big(\frac{1}{d}\Big)D\mathbf{1} = \frac{1}{d}A^\top\mathbf{1} = \frac{1}{d}A\mathbf{1} = \frac{1}{d}\begin{pmatrix} d_1 \\ \vdots \\ d_n \end{pmatrix} = \pi. \qquad \square$$

Thus the degree vector $\tilde{d} = (d_1, \ldots, d_n)^\top$ is an eigenvector of AD^{-1} with eigenvalue 1. The stationary distribution need not be unique: if G consists of two disjoint edges, then $(\frac{1}{2}, \frac{1}{2}, 0, 0)^\top$ and $(0, 0, \frac{1}{2}, \frac{1}{2})^\top$ are both stationary. If G is connected, then π is the only stationary distribution, because the eigenvalue 1 of AD^{-1} is then simple (Lemma 5.14; AD^{-1} is similar to the matrix $\tilde{A}$ studied there). In general, however, $\tilde{d}$ is not the only eigenvector of AD^{-1} whose eigenvalue has absolute value 1, and this is what makes the second question delicate.

Example 5.5. Let G be a single edge of weight 1, •——•. Then $D = I$ and $AD^{-1} = A = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$, and

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix} = -\begin{pmatrix} 1 \\ -1 \end{pmatrix},$$

so -1 is an eigenvalue of AD^{-1}. This also answers the second question negatively in general: since

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

the walk started at $p^{(0)} = (1, 0)^\top$ alternates between $(1, 0)^\top$ and $(0, 1)^\top$ and does not converge, although $\pi = (\frac{1}{2}, \frac{1}{2})^\top$ is stationary.

The graph in this example is bipartite. For connected graphs that are not bipartite, the answer to the second question is yes.

Theorem 5.6. *If G is connected and not bipartite, then for every $p^{(0)} \in \mathbb{R}^n$ with $p^{(0)} \geq 0$ and $\|p^{(0)}\|_1 = 1$,*

$$\lim_{k\to\infty} (AD^{-1})^k p^{(0)} = \pi.$$

We prove Theorem 5.6 by answering a more general question: how fast the walk approaches π, or how fast it *mixes*. The proof is completed after Lemma 5.14, and Definition 5.15 makes the speed of mixing precise.

5.3 Symmetrizing the walk

Our tool is the spectral theorem (Theorem 5.10 below). The matrix A is symmetric, but AD^{-1} is not symmetric in general, so we change variables:

$$\tilde{A} := D^{-1/2} A D^{-1/2}, \qquad \tilde{p}^{(k)} := D^{-1/2} p^{(k)}, \qquad \tilde{\pi} := D^{-1/2}\pi.$$

The matrix $\tilde{A}$ is symmetric, and $\tilde{A} = D^{-1/2}(AD^{-1})D^{1/2}$.

Claim 5.7. *Let $x \in \mathbb{R}^n$, $\lambda \in \mathbb{R}$ and $y = D^{-1/2}x$. Then $AD^{-1}x = \lambda x$ if and only if $\tilde{A}y = \lambda y$.*

Proof. By the definition of $\tilde{A}$,

$$\tilde{A}y = D^{-1/2} A D^{-1/2} D^{-1/2} x = D^{-1/2} A D^{-1} x.$$

As $D^{-1/2}$ is invertible, $AD^{-1}x = \lambda x$ holds if and only if $D^{-1/2}AD^{-1}x = \lambda D^{-1/2}x$, that is, if and only if $\tilde{A}y = \lambda y$. □

Applied to $x = \pi$, the claim gives $\tilde{A}\tilde{\pi} = \tilde{\pi}$. Moreover $\tilde{p}^{(k+1)} = D^{-1/2}AD^{-1}p^{(k)} = \tilde{A}D^{-1/2}p^{(k)} = \tilde{A}\tilde{p}^{(k)}$, so $\tilde{p}^{(k)} = \tilde{A}^k \tilde{p}^{(0)}$.

Definition 5.8 (Error)**.** For $k \geq 0$, the *error* of the walk at time k is $\varepsilon^{(k)} := \pi - p^{(k)}$, and its *scaled error* is

$$\tilde{\varepsilon}^{(k)} := D^{-1/2}\varepsilon^{(k)} = \tilde{\pi} - \tilde{A}^k \tilde{p}^{(0)} = \tilde{\pi} - \tilde{p}^{(k)}.$$

We ask how fast $\tilde{\varepsilon}^{(k)}$ tends to 0 as k grows, and whether it does at all. The scaled error obeys a much simpler recurrence than $p^{(k)}$. Since $\tilde{A}\tilde{\pi} = \tilde{\pi}$,

$$\tilde{A}\tilde{\varepsilon}^{(k)} = \tilde{A}\tilde{\pi} - \tilde{A}^{k+1}\tilde{p}^{(0)} = \tilde{\pi} - \tilde{A}^{k+1}\tilde{p}^{(0)} = \tilde{\varepsilon}^{(k+1)},$$

and therefore

$$\tilde{\varepsilon}^{(k)} = \tilde{A}^k \tilde{\varepsilon}^{(0)} \qquad \text{for every } k \geq 0. \tag{5.1}$$

Claim 5.9. *For every starting distribution $p^{(0)}$, the vector $\tilde{\varepsilon}^{(0)}$ is orthogonal to $\tilde{\pi}$, that is, $(\tilde{\pi} - \tilde{p}^{(0)})^\top \tilde{\pi} = 0$.*

Proof. We show that $\tilde{\pi}^\top \tilde{\pi} = \tilde{p}^{(0)\top}\tilde{\pi}$. First,

$$\tilde{\pi} = D^{-1/2}\pi = \begin{pmatrix} \frac{1}{\sqrt{d_1}} & & \\ & \ddots & \\ & & \frac{1}{\sqrt{d_n}} \end{pmatrix} \begin{pmatrix} d_1/d \\ \vdots \\ d_n/d \end{pmatrix} = \begin{pmatrix} \sqrt{d_1}/d \\ \vdots \\ \sqrt{d_n}/d \end{pmatrix}, \qquad \text{so} \qquad \tilde{\pi}^\top\tilde{\pi} = \sum_i \frac{d_i}{d^2} = \frac{1}{d}.$$

On the other hand, writing $p_i = p_i^{(0)}$ and using $\sum_i p_i = 1$,

$$\tilde{\pi}^\top \tilde{p}^{(0)} = \left(\frac{\sqrt{d_1}}{d}, \ldots, \frac{\sqrt{d_n}}{d}\right) \begin{pmatrix} p_1/\sqrt{d_1} \\ \vdots \\ p_n/\sqrt{d_n} \end{pmatrix} = \sum_i \frac{p_i}{d} = \frac{1}{d}.$$

□

5.4 Spectral theorem

Theorem 5.10 (Spectral theorem)**.** *Let $A \in \mathbb{R}^{n\times n}$ be symmetric. Then:*

1) *Every eigenvalue of A is real: if $Ax = \lambda x$ for some nonzero $x \in \mathbb{C}^n$, then $\lambda \in \mathbb{R}$.*

2) *If $Ax = \lambda x$ and $Ay = \mu y$ with $\lambda \neq \mu$, then $x^\top y = 0$, that is, $x \perp y$.*

3) *There is an orthonormal basis $y_1, \dots, y_n$ of $\mathbb{R}^n$ consisting of eigenvectors of A, say $Ay_i = \lambda_i y_i$, and then*
$$A = \begin{pmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} \begin{pmatrix} - & y_1^\top & - \\ & \vdots & \\ - & y_n^\top & - \end{pmatrix}.$$

4) *$A = \sum_{i=1}^n \lambda_i y_i y_i^\top$.*

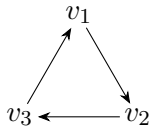
We prove the spectral theorem near the end of this lecture.

5.5 Perron–Frobenius theorem

In this subsection, A is any $n \times n$ matrix with nonnegative entries, not necessarily symmetric. Its graph $G(A)$ is the directed graph on $\{1, \dots, n\}$ with an edge from i to j whenever $A_{ij} > 0$.

Definition 5.11 (Spectral radius)**.** Let A be a square matrix and $\lambda(A)$ its set of complex eigenvalues. The *spectral radius* of A is $\rho(A) \coloneqq \max_{\lambda\in\lambda(A)} |\lambda|$, where $|z| = \sqrt{z^* z} = \sqrt{a^2 + b^2}$ is the modulus of $z = a + ib \in \mathbb{C}$.

Example 5.12. Let A be the adjacency matrix of the directed triangle:



$$A = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$

Let $\omega = e^{2\pi i/3}$, a primitive third root of unity. Since $\omega^3 = 1$,

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ \omega \\ \omega^2 \end{pmatrix} = \begin{pmatrix} \omega \\ \omega^2 \\ 1 \end{pmatrix} = \omega \begin{pmatrix} 1 \\ \omega \\ \omega^2 \end{pmatrix},$$

so ω is an eigenvalue of A. The same computation with ω replaced by ω^2 or by 1 shows that $\lambda(A) = \{1, \omega, \omega^2\}$. Thus $\rho(A) = 1$, and all three eigenvalues have absolute value $\rho(A)$; the only cycle of $G(A)$ has length 3.

Theorem 5.13 (Perron–Frobenius)**.** *Let $n \geq 2$, and let A be an $n \times n$ matrix with nonnegative entries whose graph $G(A)$ is strongly connected. Then:*

1) *$\rho(A)$ is a* simple *eigenvalue of A, and every real eigenvector x of $\rho(A)$ satisfies* $\operatorname{sign}(x_i) = \operatorname{sign}(x_j) \neq 0$ *for all i, j.*

2) *If $\theta \in \lambda(A)$ and $|\theta| = \rho(A)$, then $\theta/\rho(A)$ is a primitive mth root of unity for some integer $m \geq 1$, and every cycle in $G(A)$ has length a multiple of m.*

3) Let x be an eigenvector of $\rho(A)$ with positive entries, which exists by part 1. Then every nonnegative eigenvector of A is a positive multiple of x.

We state this theorem without proof; see, for example, [HJ12, Chapter 8]. Lecture 6 proves a symmetric version, which is all this lecture uses.

5.6 Back to the symmetric case

From now on we assume that G is connected. Let $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$ be the eigenvalues of the symmetric matrix $\tilde{A}$, listed with multiplicity.

Lemma 5.14. *Let G be connected. Then $\lambda_n = 1$, with unit eigenvector*

$$y_n = \sqrt{d}\,\tilde{\pi} = \left(\sqrt{d_1/d}, \ldots, \sqrt{d_n/d}\right)^\top;$$

moreover $\lambda_{n-1} < 1$, and $-1 \le \lambda_1$, with $-1 < \lambda_1$ if G is not bipartite.

Proof. The matrix $\tilde{A}$ is symmetric with nonnegative entries, and $\tilde{A}_{ij} > 0$ exactly when $w_{ij} > 0$. So $\tilde{A}$ is the adjacency matrix of the connected graph G with the weights $w_{ij}/\sqrt{d_i d_j}$, and the symmetric Perron–Frobenius theorem of Lecture 6 applies to it. We have $\tilde{A}\tilde{\pi} = \tilde{\pi}$, and $\tilde{\pi}^\top \tilde{\pi} = 1/d$ by the proof of Claim 5.9, so y_n is a unit eigenvector of $\tilde{A}$ with eigenvalue 1, and all its entries are positive. By part a) of the symmetric Perron–Frobenius theorem, the largest eigenvalue μ of $\tilde{A}$ has an eigenvector x with positive entries. If $\mu \ne 1$, then $x^\top y_n = 0$ by Theorem 5.10(2), which is impossible because x and y_n have positive entries. Hence $\lambda_n = \mu = 1$. Part c) of that theorem gives $\lambda_{n-1} < \lambda_n = 1$, part b) gives $\lambda_1 \ge -\lambda_n = -1$, and part d) gives $\lambda_1 = -1$ only if G is bipartite. □

Let $y_1, \ldots, y_n$ be an orthonormal basis of eigenvectors of $\tilde{A}$ with $\tilde{A}y_i = \lambda_i y_i$ and y_n as in Lemma 5.14 (Theorem 5.10(3)). By Claim 5.9, $\tilde{\varepsilon}^{(0)} \perp y_n$, so its expansion in this basis has no y_n term:

$$\tilde{\varepsilon}^{(0)} = \alpha_1 y_1 + \cdots + \alpha_{n-1} y_{n-1}, \qquad \alpha_i = y_i^\top \tilde{\varepsilon}^{(0)}.$$

By (5.1),

$$\tilde{\varepsilon}^{(1)} = \tilde{A}\tilde{\varepsilon}^{(0)} = \lambda_1\alpha_1 y_1 + \cdots + \lambda_{n-1}\alpha_{n-1} y_{n-1},$$
$$\tilde{\varepsilon}^{(k)} = \tilde{A}^k\tilde{\varepsilon}^{(0)} = \lambda_1^k\alpha_1 y_1 + \cdots + \lambda_{n-1}^k\alpha_{n-1} y_{n-1}.$$

We show that the speed of mixing is governed by

$$\lambda := \max\{|\lambda_1|, |\lambda_{n-1}|\} = \max_{i<n} |\lambda_i|.$$

By Lemma 5.14, $\lambda < 1$ whenever G is connected and not bipartite.

The error can be measured in several norms, for example $\|\cdot\|_1$, $\|\cdot\|_2$ or $\|\cdot\|_\infty$. We first bound $\|\tilde{\varepsilon}^{(k)}\|_2$, where the orthonormality of the y_i applies directly: it gives $\|\tilde{\varepsilon}^{(0)}\|_2^2 = \sum_{i<n} \alpha_i^2$ and

$$\|\tilde{\varepsilon}^{(k)}\|_2 = \sqrt{\textstyle\sum_{i<n} \lambda_i^{2k}\alpha_i^2} \le \sqrt{\textstyle\sum_{i<n} \lambda^{2k}\alpha_i^2} = \lambda^k \sqrt{\textstyle\sum_{i<n} \alpha_i^2}.$$

Hence

$$\|\tilde{\varepsilon}^{(k)}\|_2 \le \lambda^k \|\tilde{\varepsilon}^{(0)}\|_2 \qquad \text{for every } k \ge 0. \tag{5.2}$$

If G is not bipartite, then $\lambda < 1$, so we can pick k with $\lambda^k \le \frac{1}{2}$. Then $\|\tilde{\varepsilon}^{(jk)}\|_2 \le \lambda^{jk}\|\tilde{\varepsilon}^{(0)}\|_2 \le 2^{-j}\|\tilde{\varepsilon}^{(0)}\|_2$ for every $j \ge 0$: every k steps, the scaled error at least halves.

Proof of Theorem 5.6. Let G be connected and not bipartite. By Lemma 5.14, $\lambda < 1$, so $\tilde{\varepsilon}^{(k)} \to 0$ by (5.2). Hence

$$(AD^{-1})^k p^{(0)} = p^{(k)} = D^{1/2}\tilde{p}^{(k)} = D^{1/2}\big(\tilde{\pi} - \tilde{\varepsilon}^{(k)}\big) = \pi - D^{1/2}\tilde{\varepsilon}^{(k)} \longrightarrow \pi \qquad (k \to \infty). \qquad \square$$

Definition 5.15 (Mixing rate)**.** Let $\|\cdot\|$ be a norm on $\mathbb{R}^n$. The *mixing rate* of the walk on G in $\|\cdot\|$ is the smallest integer $k \geq 1$ such that $2\|\varepsilon^{(k)}\| \leq \|\varepsilon^{(0)}\|$ for every starting distribution $p^{(0)}$. We write k_1 and k_2 for the mixing rates in $\|\cdot\|_1$ and $\|\cdot\|_2$.

Remark 5.16. The bound (5.2) concerns the scaled error. Since $\varepsilon^{(k)} = D^{1/2}\tilde{\varepsilon}^{(k)}$, we have $\sqrt{\min_i d_i}\,\|\tilde{\varepsilon}^{(k)}\|_2 \leq \|\varepsilon^{(k)}\|_2 \leq \sqrt{\max_i d_i}\,\|\tilde{\varepsilon}^{(k)}\|_2$, and therefore

$$\|\varepsilon^{(k)}\|_2 \leq \sqrt{\frac{\max_i d_i}{\min_i d_i}}\,\lambda^k \|\varepsilon^{(0)}\|_2.$$

In particular, the mixing rate k_2 is at most the smallest k with $\sqrt{\max_i d_i / \min_i d_i}\,\lambda^k \leq \frac{1}{2}$.

For the scaled error the bound (5.2) is attained. Since $\pi > 0$, the vectors $\pi - p^{(0)}$, for $p^{(0)}$ a starting distribution, include every sufficiently small vector with coordinate sum 0, so $\tilde{\varepsilon}^{(0)} = D^{-1/2}(\pi - p^{(0)})$ can be a small multiple of any vector orthogonal to y_n, in particular of an eigenvector y_i with $i < n$ and $|\lambda_i| = \lambda$; then $\|\tilde{\varepsilon}^{(k)}\|_2 = \lambda^k \|\tilde{\varepsilon}^{(0)}\|_2$. Hence the mixing rate defined by the scaled error $\tilde{\varepsilon}$ in $\|\cdot\|_2$ is exactly the smallest k with $\lambda^k \leq \frac{1}{2}$.

5.7 ℓ_1 norm and Cauchy–Schwarz

Fact 5.17 (Cauchy–Schwarz)**.** *For $a, b \in \mathbb{R}^n$,*

$$(a^\top b)^2 \leq (a^\top a)(b^\top b).$$

For $a = (a_1, \ldots, a_n)^\top$, let $|a| = (|a_1|, \ldots, |a_n|)^\top$. For example, take $b = \mathbf{1}$ and $a \in \mathbb{R}^n$ arbitrary. Applying Cauchy–Schwarz to $|a|$ and b gives

$$\|a\|_1^2 = (|a|^\top b)^2 \leq |a|^\top |a| \cdot n = a^\top a \cdot n,$$

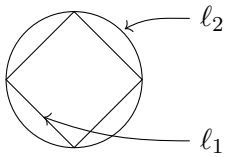
and therefore

$$\|a\|_1 \leq \sqrt{n}\,\|a\|_2. \tag{5.3}$$

Claim 5.18. *Let G be connected and not bipartite. Then the mixing rate in ℓ_1 is $O(\log n)$ times the mixing rate in ℓ_2: $k_1 = O(k_2 \log n)$. More precisely, $k_1 \leq (\lceil \log_2 n \rceil + 1)\, k_2$.*

Better estimates for the ℓ_1 error are harder to obtain.

Proof. We also need $\|a\|_2 \leq \|a\|_1$, which holds because $\|a\|_1^2 = \sum_{i,j} |a_i||a_j| \geq \sum_i a_i^2$: the ℓ_1 unit ball (the diamond) lies inside the ℓ_2 unit ball (the circle).



By Remark 5.16, $k = k_2$ is finite. The walk started at $p^{(0)}$ is, from time jk on, the walk started at the distribution $p^{(jk)}$, so Definition 5.15 gives $\|\varepsilon^{((j+1)k)}\|_2 \le \frac{1}{2}\|\varepsilon^{(jk)}\|_2$ for every $j \ge 0$: the ℓ_2 error at least halves every k steps. Let $m = \lceil \log_2 n \rceil + 1$, so that $2^{-m} \le 1/(2n)$. Using (5.3) for the first step,

$$\|\varepsilon^{(mk)}\|_1 \le \sqrt{n}\,\|\varepsilon^{(mk)}\|_2 \le 2^{-m}\sqrt{n}\,\|\varepsilon^{(0)}\|_2 \le \frac{1}{2\sqrt{n}}\|\varepsilon^{(0)}\|_2 \le \frac{1}{2}\|\varepsilon^{(0)}\|_1.$$

This holds for every starting distribution, so $k_1 \le mk_2$. Finally, $m \le 2\lceil \log_2 n \rceil$ for $n \ge 2$, so $k_1 = O(k_2 \log n)$. □

5.8 Proof of the spectral theorem

We prove Theorem 5.10. For part (1), let $Ax = \lambda x$ with $x \in \mathbb{C}^n$ nonzero, and let x^* be the conjugate transpose of x. Then $x^*Ax = \lambda\, x^*x$. The number x^*Ax is real, because A is real and symmetric and so its complex conjugate is $(x^*Ax)^* = x^*A^\top x = x^*Ax$. Since also $x^*x = \sum_i |x_i|^2 > 0$, the eigenvalue λ is real. For part (2), $\lambda\, x^\top y = (Ax)^\top y = x^\top Ay = \mu\, x^\top y$, so $(\lambda - \mu)\, x^\top y = 0$ and $x^\top y = 0$. Part (4) follows from part (3) by expanding the product of the three matrices.

It remains to prove part (3), in the following equivalent form: there exist an orthogonal matrix Q and a diagonal matrix Λ such that $A = Q\Lambda Q^\top$. (The columns of Q are then the orthonormal eigenvectors, and the diagonal of Λ holds the eigenvalues.)

Proof. We use induction on n. When $n = 1$, the matrix A is diagonal, and $Q = (1)$, $\Lambda = A$ work. Assume that it holds for $n = m$, that is, for every $m \times m$ real symmetric matrix there exist an orthogonal matrix Q and a diagonal matrix Λ with the required property. Let A be an $(m+1) \times (m+1)$ real symmetric matrix. By part (1), A has a real eigenvalue λ. Since $A - \lambda I$ is a real singular matrix, λ has an eigenvector α with real entries, which we normalize so that $\|\alpha\|_2 = 1$. By the Gram–Schmidt process, α extends to an *orthonormal* basis $q_1, \dots, q_{m+1}$ of $\mathbb{R}^{m+1}$ with $q_1 = \alpha$. Let $P := (q_1\, q_2 \cdots q_{m+1})$ and $C := P^\top AP = (c_{ij})_{(m+1)\times(m+1)}$.

We claim that $c_{11} = \lambda$ and $c_{i1} = 0$ for $i \ne 1$. Indeed, P is orthogonal, so $AP = PC$, that is, $A(q_1\, q_2 \cdots q_{m+1}) = (q_1\, q_2 \cdots q_{m+1})\, C$. Comparing first columns gives $Aq_1 = \sum_{i=1}^{m+1} c_{i1} q_i$. But $q_1 = \alpha$ is an eigenvector, so $\lambda q_1 = \sum_{i=1}^{m+1} c_{i1} q_i$. Since $q_1, \dots, q_{m+1}$ are linearly independent, $c_{11} = \lambda$ and $c_{i1} = 0$ for $i \ne 1$. So C has the block form $\left(\begin{smallmatrix} \lambda & \star \\ 0 & A' \end{smallmatrix}\right)$, where A' is the lower right $m \times m$ block. The matrix C is symmetric, because $C^\top = P^\top A^\top P = P^\top AP = C$, so $\star = 0$, and $C = \left(\begin{smallmatrix} \lambda & 0 \\ 0 & A' \end{smallmatrix}\right)$ with A' a symmetric $m \times m$ matrix. By the induction hypothesis, there exist an orthogonal matrix Q and a diagonal matrix Λ such that $A' = Q\Lambda Q^\top$. Therefore

$$C = \begin{pmatrix} \lambda & 0 \\ 0 & A' \end{pmatrix} = \begin{pmatrix} \lambda & 0 \\ 0 & Q\Lambda Q^\top \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & Q \end{pmatrix} \begin{pmatrix} \lambda & 0 \\ 0 & \Lambda \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & Q \end{pmatrix}^\top,$$

and hence

$$A = PCP^\top = P \begin{pmatrix} 1 & 0 \\ 0 & Q \end{pmatrix} \begin{pmatrix} \lambda & 0 \\ 0 & \Lambda \end{pmatrix} \left(P \begin{pmatrix} 1 & 0 \\ 0 & Q \end{pmatrix}\right)^\top.$$

The matrix $P\left(\begin{smallmatrix} 1 & 0 \\ 0 & Q \end{smallmatrix}\right)$ is a product of orthogonal matrices, hence orthogonal, which completes the induction. □

5.9 Random walks on the path graph

Let P_n be the path $v_1 v_2 \cdots v_n$ with unit weights and $n \geq 2$. The path is bipartite, so by part d) of the symmetric Perron–Frobenius theorem (Lecture 6), $\lambda_1 = -1$ for the walk above, hence $\lambda = 1$; already for $n = 2$, Example 5.5 shows a walk on P_n that does not converge. We therefore consider the lazy random walk on P_n, which at each step stays where it is with probability $\frac{1}{2}$ and otherwise moves as before. Its transition matrix is $\frac{1}{2}(I + AD^{-1})$, and π is still stationary. The analysis above applies with $\tilde{A}$ replaced by $D^{-1/2} \cdot \frac{1}{2}(I + AD^{-1}) \cdot D^{1/2} = \frac{1}{2}(I + \tilde{A})$, whose eigenvalues $\frac{1}{2}(1 + \lambda_i)$ all lie in $[0, 1]$. So the factor λ in (5.2) becomes $\lambda_{\text{lazy}} := \frac{1}{2}(1 + \lambda_{n-1})$.

We show that $\lambda_{\text{lazy}} = \lambda(P_n)$ is $1 - \Theta(1/n^2)$.

Proposition 5.19. *For the lazy random walk on P_n with $n \geq 2$,*

$$\lambda_{\text{lazy}} = \cos^2\Big(\frac{\pi}{2(n-1)}\Big), \qquad \textit{so} \qquad 1 - \frac{\pi^2}{4(n-1)^2} \leq \lambda_{\text{lazy}} \leq 1 - \frac{1}{(n-1)^2}.$$

Proof. The matrix $\tilde{A}$ has the same eigenvalues as $D^{-1}A = D^{-1/2}\tilde{A}D^{1/2}$. For $k = 0, 1, \ldots, n-1$, let $\theta = \pi k/(n-1)$ and $f(j) = \cos(\theta(j-1))$ for $1 \leq j \leq n$. For $1 < j < n$, the identity $\cos(a - \theta) + \cos(a + \theta) = 2\cos a \cos\theta$ gives $(D^{-1}Af)(j) = \frac{1}{2}(f(j-1) + f(j+1)) = \cos\theta\, f(j)$. At the ends, $(D^{-1}Af)(1) = f(2) = \cos\theta\, f(1)$ and $(D^{-1}Af)(n) = f(n-1) = \cos(\pi k - \theta) = \cos(\pi k)\cos\theta = \cos\theta\, f(n)$. So the n distinct numbers $\cos(\pi k/(n-1))$ are the eigenvalues of $\tilde{A}$, and $\lambda_{n-1} = \cos(\pi/(n-1))$. Hence $\lambda_{\text{lazy}} = \frac{1}{2}(1 + \cos(\pi/(n-1))) = \cos^2 x$ with $x = \pi/(2(n-1)) \in (0, \pi/2]$. Finally, $1 - \lambda_{\text{lazy}} = \sin^2 x$, and $2x/\pi \leq \sin x \leq x$ on $[0, \pi/2]$. □

Thus $\lambda_{\text{lazy}} \leq 1 - 1/n^2$, and $\big(1 - \frac{1}{n^2}\big)^{n^2} \leq \frac{1}{e} < \frac{1}{2}$ because $1 - x \leq e^{-x}$. By (5.2), the scaled error of the lazy walk on P_n at least halves every n^2 steps. In this sense the mixing rate of the lazy walk on P_n is $O(n^2)$. Conversely, let $n \geq 4$ and $s = \pi^2/(4(n-1)^2)$, so that $s \leq \frac{1}{2}$. On $[0, \frac{1}{2}]$ we have $1 - s \geq 4^{-s}$ (the left side is linear, the right side is convex, and they agree at 0 and $\frac{1}{2}$), so Proposition 5.19 gives $\lambda_{\text{lazy}}^k \geq (1-s)^k \geq 4^{-sk} > \frac{1}{2}$ for every $k < 2(n-1)^2/\pi^2$. By Remark 5.16, for every such k some start has a scaled error that has not halved after k steps. So the mixing rate of the lazy walk on P_n, measured by the scaled error, is $\Theta(n^2)$.

6 Perron–Frobenius and Symmetric Perron–Frobenius

Lecture of January 27, 2020. Based on Gary L. Miller's handwritten notes. (The general Perron–Frobenius theorem was stated, without proof, in Lecture 5.)

The convergence argument for random walks in Lecture 5 needs two facts about a symmetric matrix with nonnegative entries and a connected graph: its largest eigenvalue is simple, and its smallest eigenvalue equals minus the largest one only when the graph is bipartite. This lecture proves both, as parts of the Perron–Frobenius theorem for symmetric matrices. The main tools are the Rayleigh quotient and the observation that replacing a vector by its entrywise absolute value cannot decrease $x^\top Ax$.

6.1 Symmetric Perron–Frobenius

Theorem 6.1 (Symmetric Perron–Frobenius)**.** *Let $G = (V, E, w)$ be a connected undirected graph on $n \geq 2$ vertices with positive edge weights, let $A = \mathrm{Adj}(G)$, and let $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$ be the eigenvalues of A, listed with multiplicity. Then*

a) there exists $x \in \mathbb{R}^n$ such that $Ax = \mu_1 x$ and $x > 0$, that is, $x_i > 0$ for every i;

b) $\mu_1 \geq |\mu_n|$;

c) $\mu_1 > \mu_2$;

d) $\mu_1 = -\mu_n$ if and only if G is bipartite.

The matrix A is symmetric with nonnegative entries, and $A_{ij} > 0$ exactly when $(i, j) \in E$; its eigenvalues are real by the spectral theorem (Lecture 5). Throughout the lecture, G, A and the μ_i are as in the theorem. Part d) relies on a second graph built from A.

Definition 6.2. The graph G^2 is the graph of A^2: its vertex set is V, and distinct vertices i and j are adjacent in G^2 if $(A^2)_{ij} > 0$, that is, if i and j have a common neighbor in G. More generally, G^k denotes the graph of A^k.

Lemma 6.3. *Let G be a connected undirected graph on $n \geq 2$ vertices. Then G is bipartite if and only if G^2 is not connected.*

Proof. Suppose first that G is bipartite, with parts X and Y. Both parts are nonempty, since G is connected with $n \geq 2$ and so has an edge. Two vertices with a common neighbor lie in the same part, so G^2 has no edge between X and Y and is not connected.

Conversely, suppose that G is not bipartite. Then G has a closed walk C of odd length. (Otherwise, for a fixed vertex r, all walks from r to a vertex v would have lengths of the same parity, since one followed by the reverse of another is a closed walk; sorting the vertices by this parity would give a bipartition.) Let z be a vertex of C and let $u, v \in V$. If a walk P from u to v has odd length, then going from u to z, around C, back to u the same way, and then along P gives a walk of even length. So there is a walk $u = w_0, w_1, \ldots, w_{2s} = v$. Consecutive vertices w_{2r} and w_{2r+2} have the common neighbor w_{2r+1}, so they are equal or adjacent in G^2. Hence G^2 is connected. □

Proof of a)

For $x \in \mathbb{R}^n$, let $|x|$ denote the vector with entries $|x|_i = |x_i|$. We use the following characterization of μ_1.

Lemma 6.4. *Let A be a real symmetric matrix with largest eigenvalue μ_1. Then*

$$\mu_1 = \max_{x \neq 0} \frac{x^\top A x}{x^\top x},$$

and a nonzero vector z attains this maximum if and only if $Az = \mu_1 z$.

Proof. By the spectral theorem, A has an orthonormal basis of eigenvectors $u_1, \dots, u_n$ with $Au_i = \mu_i u_i$. Write a nonzero z as $z = \sum_i c_i u_i$. Then

$$\frac{z^\top A z}{z^\top z} = \frac{\sum_i \mu_i c_i^2}{\sum_i c_i^2} \leq \mu_1,$$

with equality if and only if $c_i = 0$ whenever $\mu_i < \mu_1$, that is, if and only if z lies in the eigenspace of μ_1. Equality holds for $z = u_1$. □

Let $x \neq 0$ and compare the Rayleigh quotients of x and $|x|$.

1) $|x|^\top |x| = \sum_i |x_i| \cdot |x_i| = \sum_i x_i^2 = x^\top x$.

2) $|x|^\top A |x| = \sum_{i,j} A_{ij} |x_i| |x_j| \geq \sum_{i,j} A_{ij} x_i x_j = x^\top A x$, since $A_{ij} \geq 0$ and $|x_i||x_j| \geq x_i x_j$.

By these two facts and Lemma 6.4,

$$\frac{x^\top A x}{x^\top x} \leq \frac{|x|^\top A |x|}{|x|^\top |x|} \leq \mu_1. \tag{6.1}$$

Now take x to be an eigenvector of μ_1. The left side of (6.1) is then μ_1, so $|x|$ attains the maximum in Lemma 6.4. Therefore $A|x| = \mu_1 |x|$, and $|x| \geq 0$ is an eigenvector of μ_1.

It remains to show that every nonnegative eigenvector z of μ_1 is positive. Suppose that $z_i = 0$ for some i. Then

$$0 = \mu_1 z_i = (Az)_i = \sum_j A_{ij} z_j,$$

a sum of nonnegative terms, so $z_j = 0$ for every neighbor j of i. Hence the set $\{i : z_i = 0\}$ is nonempty and contains every neighbor of each of its elements. Since G is connected, it is all of V, so $z = 0$, a contradiction. Taking $z = |x|$ proves a). We record what the last argument shows:

$$\text{every nonnegative eigenvector of } \mu_1 \text{ has positive entries.} \tag{6.2}$$

Proof of b)

Suppose that $Ay = \mu_n y$ and $y^\top y = 1$. By the triangle inequality and (6.1),

$$|\mu_n| = |y^\top A y| = \left| \sum_{i,j} A_{ij} y_i y_j \right| \leq |y|^\top A |y| \leq \mu_1.$$

Proof of c)

Let x be a positive eigenvector of μ_1, which exists by a), normalized so that $x^\top x = 1$. By the spectral theorem, we can pick a unit eigenvector y of μ_2 with $x^\top y = 0$. Since $x > 0$ and $y \neq 0$, the condition $x^\top y = \sum_i x_i y_i = 0$ forces y to have a negative entry and a positive entry. Now

$$\mu_2 = y^\top A y \overset{(*)}{\leq} |y|^\top A |y| \leq \mu_1. \tag{6.3}$$

Claim 6.5. *If no entry of y is zero, then the inequality $(*)$ is strict.*

Proof. Split the sum according to the sign of $y_i y_j$:

$$y^\top A y = \sum_{y_i y_j < 0} A_{ij} y_i y_j + \sum_{y_i y_j \geq 0} A_{ij} y_i y_j.$$

The second sum equals $\sum_{y_i y_j \geq 0} A_{ij} |y_i||y_j|$, and

$$\sum_{y_i y_j < 0} A_{ij} y_i y_j < \sum_{y_i y_j < 0} A_{ij} |y_i||y_j|$$

as soon as some pair (i, j) with $A_{ij} > 0$ has $y_i y_j < 0$. Such a pair exists. The sets $\{i : y_i > 0\}$ and $\{i : y_i < 0\}$ are nonempty and, because no entry of y is zero, they partition V. Since G is connected, some edge (i, j) joins the two sets, and then $A_{ij} > 0$ and $y_i y_j < 0$. □

We now prove c) in two cases.

Case 1: $y_j \neq 0$ for every j, that is, $|y| > 0$. By Claim 6.5 and (6.3), $\mu_2 < |y|^\top A |y| \leq \mu_1$.

Case 2: $y_i = 0$ for some i. Suppose that $\mu_1 = \mu_2$. Then $Ay = \mu_1 y$, so y is an eigenvector of μ_1. By (6.1) applied to y, the vector $|y|$ also attains the maximum in Lemma 6.4, so it is an eigenvector of μ_1. It is nonnegative, so $|y| > 0$ by (6.2). This contradicts $|y|_i = 0$. Hence $\mu_1 > \mu_2$.

Remark 6.6. In Case 2 the inequality $(*)$ itself can be an equality, which is why Claim 6.5 assumes that y has no zero entry. For the star $K_{1,3}$ with center v_1, leaves v_2, v_3, v_4 and unit weights, the eigenvalues are $\sqrt{3}, 0, 0, -\sqrt{3}$, and $y = (0, 1, -1, 0)^\top / \sqrt{2}$ is a unit eigenvector of $\mu_2 = 0$ orthogonal to the positive eigenvector. Here $y^\top A y = 0 = |y|^\top A |y|$, because every edge has the endpoint v_1, where y vanishes.

Proof of d)

($\Leftarrow$) Suppose that G is bipartite. By b), every eigenvalue of A is at least $-\mu_1$, so it suffices to show that $-\mu_1$ is an eigenvalue of A.

Homework 6.7. Let G be a connected bipartite graph with positive edge weights and $A = \mathrm{Adj}(G)$. Construct $y \neq 0$ such that $Ay = -\mu_1 y$.

($\Rightarrow$) Suppose that $\mu_n = -\mu_1$. By the spectral theorem, there are vectors x and y with

$$Ax = \mu_1 x, \qquad Ay = -\mu_1 y, \qquad x^\top x = y^\top y = 1, \qquad x^\top y = 0.$$

For every $k \geq 1$, if $Az = \mu z$ then $A^k z = \mu^k z$. So if $A = U\Lambda U^\top$, then $A^k = U\Lambda^k U^\top$: the eigenvalues of A^k, the matrix of G^k, are $\mu_1^k, \ldots, \mu_n^k$, and every eigenvector of A is an eigenvector of A^k. In particular,

$$A^2 x = \mu_1^2 x, \qquad A^2 y = \mu_1^2 y, \qquad x^\top y = 0,$$

and this implies that G^2 is not connected. Indeed, by b) the largest eigenvalue of A^2 is $\max_i \mu_i^2 = \mu_1^2$, and by the display it has multiplicity at least 2. The matrix A^2 is symmetric with nonnegative entries, and the proofs of a)–c) above never use the diagonal entries of the matrix, so c) holds for A^2 whenever its graph G^2 is connected. If G^2 were connected, c) would make μ_1^2 a simple eigenvalue of A^2, a contradiction. Hence G is bipartite by Lemma 6.3.

7 Differential Equations, Matrix Exponentials and Laplacians

Lecture of January 29, 2020. Based on Gary L. Miller's handwritten notes.

Graph Laplacians govern differential equations on graphs. This lecture treats two physical models. In heat diffusion the temperatures satisfy $du/dt = -Lu$, which we solve with the matrix exponential e^{-Lt}. In a system of springs and masses, Newton's law reads $-Lu = M\, d^2u/dt^2$, and its oscillating solutions come from the generalized eigenvalue problem $Lx = \lambda Mx$. Between the two models we show that a random walk that moves with small probability ε per step tends, as $\varepsilon \to 0$, to a matrix exponential (the ultra lazy walk); after the change of variables of Lecture 5, it is the exponential $e^{-\tilde{L}t}$ of the normalized Laplacian. The lecture closes by observing that when every mass equals the degree, the spring-mass system is governed by the same normalized Laplacian.

7.1 Differential equations on graphs

Our first example is diffusion. Let $G = (V, E, C)$ be an undirected graph with $V = \{v_1, \ldots, v_n\}$, m edges and thermal conductances $C\colon E \to \mathbb{R}^+$, which we also write as the diagonal $m \times m$ matrix C. Let $u = (u_1, \ldots, u_n)^\top$ be the vector of temperatures at the vertices. Orient each edge, and let B be the $m \times n$ matrix whose row for the edge e from v_i to v_j has $+1$ in column i, -1 in column j, and zeros elsewhere. Then

$$\begin{aligned} Bu &= \text{the temperature drop across each edge,} \\ CBu &= \text{the thermal flow along each edge,} \\ -B^\top CBu &= \text{the change in temperature at the vertices.} \end{aligned}$$

Indeed, $(Bu)_e = u_i - u_j$; heat flows from hot to cold at a rate proportional to the drop; and $(B^\top f)_k$ is the net flow out of v_k for a flow f on the edges, so $-B^\top CBu$ is the net flow into each vertex. This B has one row per edge, so it is the transpose of the boundary operator of Lecture 2, and $L = B^\top CB$ is the Laplacian of G. Our differential equation is

$$\frac{du}{dt} = -Lu, \qquad \text{where} \qquad \frac{du}{dt} := \begin{pmatrix} \frac{du_1}{dt} \\ \vdots \\ \frac{du_n}{dt} \end{pmatrix}. \tag{7.1}$$

Our goal is to find $u(t)$ for $t \geq 0$, given the initial temperatures $u(0) = u^{(0)}$.

We use matrix exponentials, which extend the series $e^x = 1 + x + \frac{x^2}{2!} + \cdots$ to matrices.

Definition 7.1. For a real symmetric matrix A, the *matrix exponential* is

$$e^A := I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots.$$

Let $A = U\Lambda U^\top$ with U orthogonal and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, as given by the spectral theorem (Lecture 5). Then $A^k = U\Lambda^k U^\top$ for every k, so every partial sum of the series equals $U\big(\sum_{k \leq N} \Lambda^k/k!\big)U^\top$. Letting $N \to \infty$ shows that the series converges and that $e^A = Ue^\Lambda U^\top$, that is,

$$e^A = U \begin{pmatrix} e^{\lambda_1} & & \\ & \ddots & \\ & & e^{\lambda_n} \end{pmatrix} U^\top.$$

For a scalar $t \in \mathbb{C}$, the same series defines

$$e^{At} = I + At + \frac{A^2t^2}{2!} + \cdots = U \operatorname{diag}(e^{\lambda_1 t}, \ldots, e^{\lambda_n t}) U^\top.$$

Each entry of this series is a power series in t that converges for every t, so for real t we may differentiate term by term:

$$\begin{aligned} \frac{d(e^{At})}{dt} &= A + \frac{2A^2t}{2!} + \frac{3A^3t^2}{3!} + \cdots \\ &= A(I + At + \cdots) \\ &= Ae^{At}. \end{aligned}$$

Claim 7.2. *For every $u^{(0)} \in \mathbb{R}^n$, the function $u(t) = e^{-Lt}u^{(0)}$ satisfies $u(0) = u^{(0)}$ and solves (7.1).*

Proof. First, $u(0) = e^0 u^{(0)} = Iu^{(0)} = u^{(0)}$. Second, by the derivative formula with $A = -L$,

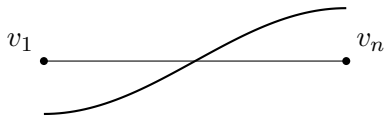
$$\frac{d(u(t))}{dt} = \frac{d(e^{-Lt})}{dt}\, u^{(0)} = -L(e^{-Lt})\, u^{(0)} = -Lu(t). \qquad \square$$

The eigenvectors of L show how the solution behaves. Write $L = U\Lambda U^\top$, where the columns of $U = (x_1 \cdots x_n)$ are orthonormal eigenvectors of L with eigenvalues $0 = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$ (Lecture 2). Let $u^{(0)} = c_1 x_1 + \cdots + c_n x_n$. Since $e^{-Lt}x_i = e^{-\lambda_i t}x_i$,

$$u(t) = e^{-\lambda_1 t}c_1 x_1 + \cdots + e^{-\lambda_n t}c_n x_n.$$

The component along x_i decays at the rate λ_i.

Example 7.3. Let $G = P_n$, the path $v_1 v_2 \cdots v_n$ with unit conductances and $n \ge 2$. Here $x_1 = \frac{1}{\sqrt{n}}(1, \ldots, 1)^\top$ with $\lambda_1 = 0$, so the component along x_1 does not decay; it is the average temperature. The second eigenvector x_2 changes sign once, like half a period of a cosine:

$x_2 =$ 

In fact, for $k = 0, 1, \ldots, n-1$, the vector with entries $\cos(\pi k(j - \frac{1}{2})/n)$, $1 \le j \le n$, is an eigenvector of L with eigenvalue $2(1 - \cos(\pi k/n))$, as a direct check of the eigenvalue equation shows, using the identity $\cos(a - \theta) + \cos(a + \theta) = 2\cos a \cos\theta$. These n eigenvalues are distinct, so x_2 is proportional to the vector with $k = 1$, and $\lambda_2 = 2(1 - \cos(\pi/n)) = 4\sin^2(\pi/(2n))$. Since $2y/\pi \le \sin y \le y$ for $0 \le y \le \pi/2$, we get $4/n^2 \le \lambda_2 \le \pi^2/n^2$. So λ_2 is of order $1/n^2$, and the component along x_2 decays like $e^{-\lambda_2 t}$. Its half-life, the time $\ln 2/\lambda_2$ after which it has halved, is of order n^2.

7.2 Ultra lazy random walks

In this subsection, $G = (V, E, w)$ has adjacency matrix A and invertible degree matrix D (every vertex has positive degree), the random walk on G has transition matrix $A^\top D^{-1}$ (Lecture 5), and $\tilde{L} := I - D^{-1/2}AD^{-1/2}$ is the *normalized Laplacian*. For $0 < \varepsilon \le 1$, the *simple lazy walk* stays where it is with probability $1 - \varepsilon$ and otherwise takes a step of the random walk; its transition matrix is

$$(1 - \varepsilon)I + \varepsilon A^\top D^{-1}.$$

Definition 7.4 (t-step walk)**.** Let $t > 0$ and $0 < \varepsilon \le 1$ with t/ε an integer. The *t-step walk* is

$$\big((1-\varepsilon)I + \varepsilon A^\top D^{-1}\big)^{t/\varepsilon},$$

that is, t/ε steps of the lazy walk, during which the walk moves t times on average.

We change variables as in Lecture 5. Conjugation by $D^{-1/2}$ commutes with taking powers, and $D^{-1/2}A^\top D^{-1}D^{1/2} = D^{-1/2}AD^{-1/2}$ because A is symmetric. Hence

$$\begin{aligned} &D^{-1/2}\big((1-\varepsilon)I + \varepsilon A^\top D^{-1}\big)^{t/\varepsilon} D^{1/2} \\ &\quad = \big((1-\varepsilon)I + \varepsilon D^{-1/2}AD^{-1/2}\big)^{t/\varepsilon} \\ &\quad = \big(I - \varepsilon(I - D^{-1/2}AD^{-1/2})\big)^{t/\varepsilon} \\ &\quad = (I - \varepsilon\tilde{L})^{t/\varepsilon}. \end{aligned}$$

Claim 7.5. *For every $t > 0$,*

$$\lim_{\varepsilon\to 0}(I - \varepsilon\tilde{L})^{t/\varepsilon} = e^{-\tilde{L}t},$$

where ε runs through the values t/N with N a positive integer.

Proof. Write $\tilde{L} = U\Lambda U^\top$ with U orthogonal and $\Lambda = \operatorname{diag}(\lambda_1,\dots,\lambda_n)$. Then $(I-\varepsilon\tilde{L})^{t/\varepsilon} = U\operatorname{diag}\big((1-\varepsilon\lambda_i)^{t/\varepsilon}\big)U^\top$. For $\varepsilon = t/N$ and each fixed i, $(1-\varepsilon\lambda_i)^{t/\varepsilon} = (1-\lambda_i t/N)^N \to e^{-\lambda_i t}$ as $N\to\infty$. Hence the limit is $U\operatorname{diag}(e^{-\lambda_i t})U^\top = e^{-\tilde{L}t}$. □

Definition 7.6 (Ultra lazy walk)**.** The *ultra lazy walk* is the family of limits of the t-step walks,

$$\lim_{\varepsilon\to 0}\big((1-\varepsilon)I + \varepsilon A^\top D^{-1}\big)^{t/\varepsilon} = D^{1/2}e^{-\tilde{L}t}D^{-1/2} = e^{-LD^{-1}t}, \qquad t \ge 0,$$

where $L = D - A$ is the Laplacian of G. In the scaled variables $\tilde{p} = D^{-1/2}p$ it is the family $e^{-\tilde{L}t}$.

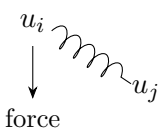
The first equality follows from the change of variables above and Claim 7.5, and the second from $D^{1/2}\tilde{L}D^{-1/2} = I - AD^{-1} = LD^{-1}$. As a limit of column-stochastic matrices, $e^{-LD^{-1}t}$ is column-stochastic. The matrix $e^{-\tilde{L}t}$ need not be: for the path P_3 and $t = 1.3$ its column sums are about 0.864, 1.192, 0.864.

7.3 Springs and graph Laplacians

Our second model is a mattress: a graph of springs $G = (V, E, k)$, where $k_{ij} > 0$ is the spring constant of the edge E_{ij} and $m_i > 0$ is the mass of the vertex v_i. We consider only vertical displacements: for example, G is embedded in $\mathbb{R}^2$ and every vertex moves only in the z-direction. Let $u_i = u_i(t)$ denote the displacement of v_i. We look for solutions of Newton's equation $F = Ma$, so we first determine the force F, the mass M and the acceleration a.

The force. Assume a linear spring model. For a displacement $u = (u_1,\dots,u_n)^\top$, the force on u_i exerted by the spring E_{ij} is

$$-(u_i - u_j)k_{ij}$$



so it pulls u_i toward u_j. Summing over the springs at v_i, the total force on u_i is $-\sum_j k_{ij}(u_i - u_j) = -(Lu)_i$, where $L = L(G)$ is the Laplacian with weights k_{ij}. Hence the force vector is $-Lu$.

The mass. The masses form the diagonal matrix

$$M = \begin{pmatrix} m_1 & & 0 \\ & \ddots & \\ 0 & & m_n \end{pmatrix}.$$

The acceleration. The acceleration is the second derivative of the displacement:

$$a = \begin{pmatrix} \frac{d^2u_1}{dt^2} \\ \vdots \\ \frac{d^2u_n}{dt^2} \end{pmatrix} = \frac{d^2u}{dt^2}.$$

Thus Newton's equation is

$$-Lu = M\Big(\frac{d^2u}{dt^2}\Big). \tag{7.2}$$

There are two ways to approach (7.2):

1) solve for u given initial conditions, say $u(0)$ and $u'(0)$;

2) find steady-state solutions, in which every vertex oscillates with one common frequency.

We do 2) by guess and check. Recall that $e^{i\omega t} = \cos\omega t + i\sin\omega t$, so

$$\begin{aligned} \frac{de^{i\omega t}}{dt} = i\omega e^{i\omega t} &= -\omega\sin\omega t + i\omega\cos\omega t \\ &= i\omega(\cos\omega t + i\sin\omega t), \end{aligned}$$

and

$$\begin{aligned} \frac{d^2e^{i\omega t}}{dt^2} = \frac{d(i\omega e^{i\omega t})}{dt} &= i^2\omega^2 e^{i\omega t} \\ &= -\omega^2 e^{i\omega t} = -\omega^2(\cos\omega t + i\sin\omega t). \end{aligned}$$

We guess a solution of the form $u(t) = e^{\pm i\omega t}x$ for some nonzero vector $x \in \mathbb{R}^n$ and some $\omega \in \mathbb{R}$, and check for which (ω, x) it works. Since $\frac{d^2}{dt^2}(e^{\pm i\omega t}x) = -\omega^2 e^{\pm i\omega t}x$ and $e^{\pm i\omega t} \neq 0$,

$$-L(e^{\pm i\omega t}x) = M(-\omega^2 e^{\pm i\omega t}x) \quad \text{if and only if} \quad Lx = \omega^2 Mx.$$

So we need $Lx = \lambda Mx$ with $\lambda = \omega^2 \geq 0$, a *generalized eigenvalue problem.*

Claim 7.7. *Let M be diagonal with positive diagonal entries. If $Lx = \lambda Mx$ for some nonzero x, then λ is real and nonnegative.*

Proof. Change variables: $y = M^{1/2}x$, or $x = M^{-1/2}y$. Multiplying by $M^{-1/2}$ shows that

$$Lx = \lambda Mx \quad \text{if and only if} \quad M^{-1/2}LM^{-1/2}y = \lambda y.$$

The matrix $M^{-1/2}LM^{-1/2}$ is symmetric, and it is positive semidefinite: for every z,

$$z^\top M^{-1/2}LM^{-1/2}z = (M^{-1/2}z)^\top L\,(M^{-1/2}z) \geq 0$$

because L is positive semidefinite (Lecture 2). Hence its eigenvalues are real and nonnegative. □

Let $y_1, \ldots, y_n$ be orthonormal eigenvectors of $M^{-1/2} L M^{-1/2}$ with eigenvalues $\lambda_1, \ldots, \lambda_n$, and let $x_i = M^{-1/2} y_i$. Then $L x_i = \lambda_i M x_i$ and $x_i^\top M x_j = y_i^\top y_j = \delta_{ij}$, where δ_{ij} is 1 if $i = j$ and 0 otherwise.

Thus we have found n solutions

$$e^{i\sqrt{\lambda_j}\, t} x_j \qquad \text{for } j = 1, \ldots, n.$$

Since (7.2) is linear, linear combinations of solutions are also solutions. Let $U = (x_1 \cdots x_n)$ and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Then

$$U \begin{pmatrix} e^{i\sqrt{\lambda_1}\, t} & & \\ & \ddots & \\ & & e^{i\sqrt{\lambda_n}\, t} \end{pmatrix} c = U e^{i\Lambda^{1/2} t} c$$

is a solution for every $c \in \mathbb{C}^n$. The relations $x_i^\top M x_j = \delta_{ij}$ say that $U^\top M U = I$, so $U^{-1} = U^\top M$, and $u(t) = U e^{i\Lambda^{1/2} t} U^\top M x_0$ is a solution with $u(0) = x_0$. When $M = I$, the matrix U is orthogonal, and this solution is $U e^{i\Lambda^{1/2} t} U^\top x_0 = e^{iL^{1/2} t} x_0$, where $L^{1/2} = U \Lambda^{1/2} U^\top$. It is the solution with $u(0) = x_0$ and $u'(0) = i L^{1/2} x_0$. The equation is of second order, so $u(0)$ alone does not determine u: for example, $\cos(L^{1/2} t)\, x_0 = \frac{1}{2}(e^{iL^{1/2} t} + e^{-iL^{1/2} t}) x_0$ is the solution with $u(0) = x_0$ and $u'(0) = 0$.

7.4 The normalized Laplacian and spring-mass systems

Consider the case in which the mass of every vertex equals its degree, $m_i = d_i = \sum_j k_{ij}$, that is, $M = D$. Then the eigenproblem of the spring-mass system is

$$L x = \lambda D x,$$

and its symmetric version, from the proof of Claim 7.7, is the eigenproblem of

$$D^{-1/2} L D^{-1/2} = D^{-1/2}(D - A) D^{-1/2} = I - D^{-1/2} A D^{-1/2} = \tilde{L},$$

the normalized Laplacian that also governs the ultra lazy walk.

8 Graph Sparsifiers and the Ahlswede–Winter Theorem

Lecture of February 3, 2020. Based on Gary L. Miller's handwritten notes. Zichao Dong's scribe notes for this lecture were used as a reading aid.

A sparsifier of a dense graph G is a graph on the same vertices with far fewer edges that approximates G. This lecture defines two notions of approximation, cut preserving and spectral preserving, and builds sparsifiers by sampling edges at random with replacement. The analysis rests on the Ahlswede–Winter theorem, a matrix analogue of the Chernoff bound, which we state without proof. Sampling edges uniformly can require about as many samples as G has edges. Sampling each edge with probability proportional to its weight times its effective resistance (Lectures 2 and 3) instead gives a spectral sparsifier with $O(n \log n)$ edges; this is the theorem of Spielman and Srivastava. We also show that upper estimates of the effective resistances suffice, at the cost of more samples.

8.1 Graph sparsifiers

Throughout this lecture, graphs are undirected with positive edge weights, and L_G denotes the Laplacian of G (Lecture 2).

Given a graph $G = (V, E, w)$ with many more edges than vertices, $|E| \gg |V|$, we want a graph $H = (V, E', w')$ on the same vertex set with far fewer edges, ideally $|E'|$ close to $|V|$, such that H is similar to G. The following two definitions make similarity precise. Fix $0 < \varepsilon < 1$.

Definition 8.1. The graph H is *cut preserving* for G if for every $S \subseteq V$,

$$(1-\varepsilon) \sum_{i \in S,\ j \notin S} w_{ij} \le \sum_{i \in S,\ j \notin S} w'_{ij} \le (1+\varepsilon) \sum_{i \in S,\ j \notin S} w_{ij}.$$

Definition 8.2 (Loewner order)**.** Let A and B be symmetric positive semidefinite $n \times n$ matrices. We write $A \preceq B$ if $x^\top A x \le x^\top B x$ for every $x \in \mathbb{R}^n$.

Definition 8.3. The graph H is *spectral preserving* for G if

$$(1-\varepsilon) L_G \preceq L_H \preceq (1+\varepsilon) L_G. \tag{8.1}$$

Homework 8.4. Show that if H is spectral preserving for G, then H is cut preserving for G, with the same ε.

8.2 Generating a sparse subgraph by edge sampling

Let $G = (V, E, w)$ have edges $e_1, \dots, e_m$ with weights $w_1, \dots, w_m$. The idea is to pick a probability for each edge,

edge	e_1	$\cdots$	e_m
probability	p_1	$\cdots$	p_m

where $\sum_i p_i = 1$ and $p_i \ge 0$,

and to sample edges according to these probabilities.

Algorithm 8.5 (Sample)**.**

1) Pick an index i with probability p_i.

2) Return $\frac{1}{p_i} e_i$, that is, the graph on V whose only edge is e_i, with weight w_i/p_i.

Algorithm 8.6 (k-Sample)**.**

1) Run Sample k times independently (with replacement), obtaining $\text{Sample}_1, \ldots, \text{Sample}_k$.

2) Let $H = \text{Sample}_1 + \cdots + \text{Sample}_k$, the graph on V whose edges are the sampled edges; an edge sampled several times gets the sum of the weights of its copies.

3) Return H/k, the graph H with every weight divided by k.

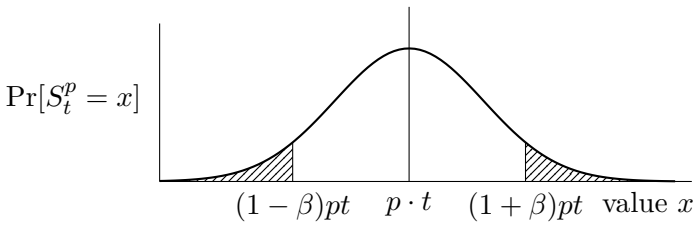
The question is which probabilities $p_1, \ldots, p_m$ and which k give good sparsifiers. We view a sample as a Laplacian matrix, the Laplacian of a graph with one edge, and the output of k-Sample as $1/k$ times a sum of k such Laplacians: $L_{H/k} = \frac{1}{k} \sum_{j=1}^{k} L_{\text{Sample}_j}$.

8.3 Chernoff bounds

The analysis of k-Sample uses a matrix version of the Chernoff bound. We first recall the scalar version. Let $X_1, \ldots, X_t$ be independent 0/1 random variables with $\Pr[X_i = 1] = p$, or equivalently $\mathbb{E}[X_i] = p$. The binomial random variable is

$$S_t^p = X_1 + \cdots + X_t,$$

and by linearity of expectation, $\mathbb{E}[S_t^p] = \sum_{i=1}^{t} \mathbb{E}[X_i] = p \cdot t$. The Chernoff bounds say that the two tails of the distribution of S_t^p, shaded below, have small probability.



Theorem 8.7 (Chernoff bounds)**.** *Let S_t^p be as above. For every $0 \le \beta < 1$,*

$$\Pr\left[S_t^p < (1-\beta)pt\right] < e^{-\beta^2 pt/2},$$
$$\Pr\left[S_t^p > (1+\beta)pt\right] < e^{-\beta^2 pt/3}.$$

We state these bounds without proof.

8.4 Ahlswede–Winter theorem

The Ahlswede–Winter theorem, which we call AW, is the analogue of the Chernoff bounds for sums of random positive semidefinite matrices.

Theorem 8.8 (Ahlswede–Winter [AW02])**.** *Let $Y_1, \ldots, Y_k$ be independent, identically distributed random symmetric positive semidefinite (SPSD) $n \times n$ matrices with $\mathbb{E}[Y_i] = Z$, and let $\mu > 0$ be such that $Y_i \preceq \mu Z$ for every possible value of Y_i. Then for every $0 < \varepsilon \le 1$,*

$$\Pr\Big[(1-\varepsilon)Z \preceq \frac{1}{k}\sum_{i=1}^{k} Y_i \preceq (1+\varepsilon)Z\Big] \ge 1 - 2ne^{-c\varepsilon^2 k/\mu},$$

where the constant $c = 1/4$ works.

We state AW without proof; Lecture 11 derives a version of it from Tropp's matrix Chernoff bound. We now do some examples.

Example 8.9. Set $Z = I^{n\times n}$, and let E_i be the $n \times n$ matrix whose only nonzero entry is a 1 in position (i, i):

$$E_i = \begin{pmatrix} 0 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 0 \end{pmatrix}.$$

Define the random matrix Y by picking $i \in \{1, \dots, n\}$ uniformly at random and returning nE_i. Then

$$\mathbb{E}[Y] = \sum_{i=1}^{n} \frac{1}{n}\, nE_i = \sum_i E_i = I.$$

Moreover $nE_i \preceq nI$, since $x^\top (nE_i)x = nx_i^2 \le nx^\top x$, so $\mu = n$ works.

We pick k so that the failure probability $2ne^{-c\varepsilon^2 k/n}$ in AW is small. Try $k = c' n \log n$ for a constant c', with natural logarithms:

$$2ne^{-c\varepsilon^2 k/n} = 2n(e^{\log n})^{-c'c\varepsilon^2} = 2n \cdot n^{-c'c\varepsilon^2}.$$

Setting $c'c\varepsilon^2 = 2$, that is, $c' = 2/(c\varepsilon^2)$, gives $2n/n^2 = 2/n$. So with $k \ge 2n \log n/(c\varepsilon^2) = 8n \log n/\varepsilon^2$ samples, $(1-\varepsilon)I \preceq \frac{1}{k}\sum_i Y_i \preceq (1+\varepsilon)I$ holds with probability at least $1 - 2/n$.

Remark 8.10. The restriction $\varepsilon \le 1$ in AW cannot be dropped. In Example 8.9 with $n = 100$, $k = 1$ and $\varepsilon = 50$, the single sample $Y_1 = 100E_i$ has the eigenvalue $100 > 1+\varepsilon$, so the event in AW never occurs, while the bound would give probability at least $1 - 200e^{-2500/400} = 1 - 200e^{-6.25} > 0.6$.

For an edge $e = (i, j)$, let $H_e = H_{ij}$ be the Laplacian of the graph on V whose only edge is e, with weight 1:

$$H_{ij} = \begin{pmatrix} \ddots & & & \\ & 1 & -1 & \\ & -1 & 1 & \\ & & & \ddots \end{pmatrix} = H_e,$$

with entries 1 in positions (i, i) and (j, j), entries -1 in positions (i, j) and (j, i), and zeros elsewhere. Then

$$L_G = \sum_{e\in E} w_e H_e.$$

Whether $H_e \preceq \alpha L_G$ depends on the effective resistance ER_e between the endpoints of e.

Homework 8.11. Let G be connected, let $e \in E$, and let $\alpha \ge 0$. Show that $H_e \preceq \alpha L_G$ if and only if $\alpha \ge \mathrm{ER}_e$. (Lecture 9 proves the forward implication.)

Example 8.12. Let $Z = L_G$. Assume that G is connected and that $w_e = 1$ for every edge, and pick each edge uniformly: define

$$Y \coloneqq mH_e \quad \text{for } e \in E \text{ chosen uniformly at random.}$$

Then

$$\mathbb{E}[Y] = \sum_{e \in E} \frac{1}{m}\, m H_e = \sum_{e \in E} H_e = L_G.$$

AW needs $mH_e \preceq \mu L_G$ for every e, that is, $H_e \preceq \frac{\mu}{m} L_G$. By Homework 8.11, this holds if and only if $\mathrm{ER}_e \le \frac{\mu}{m}$, that is, $m \,\mathrm{ER}_e \le \mu$.

If e is an isthmus (a bridge), then $\mathrm{ER}_e = 1$: a unit current between the endpoints of e must flow entirely through e. Thus if G has an isthmus, we need $\mu \ge m$. With $\mu = m$, the bound in AW is $2ne^{-\varepsilon^2 k/(4m)}$, which by the computation of Example 8.9 is at most $2/n$ once $k \ge 8m \log n/\varepsilon^2$: that is $O(m \log m)$ samples for fixed ε. Each edge is then sampled $k/m = \Omega(\log n)$ times on average, so sampling gives back essentially all of G, and H is not sparse.

8.5 Picking edges by effective resistance

Picking edges uniformly performs badly, because a single edge of large effective resistance forces μ to be large. We instead pick each edge with probability proportional to $w(e)\,\mathrm{ER}_e$. The normalizing constant is the sum of these numbers.

Claim 8.13. *If G is connected and has unit weights, then $\sum_{e \in E} \mathrm{ER}_e = n - 1$.*

For a tree the claim is immediate: each of its $n - 1$ edges is an isthmus, with effective resistance 1. In general it is the case of unit weights of the following theorem.

Theorem 8.14 (Foster's theorem)**.** *If $G = (V, E, w)$ is connected and has n vertices, then*

$$\sum_{e \in E} w(e)\,\mathrm{ER}_e = n - 1.$$

Proof. For an edge $e = (a, b)$, let $\chi_e = \chi_a - \chi_b$, where χ_a and χ_b are the characteristic vectors of the vertices a and b, so that $H_e = \chi_e \chi_e^\top$. By the definition of effective resistance (Lecture 2), $\mathrm{ER}_e = v_a - v_b = \chi_e^\top v$, where v is a vector of voltages that sends a unit current from a to b, that is, $L_G v = \chi_e$. Since G is connected, $\mathrm{Ker}(L_G)$ is spanned by $\mathbf{1}$ (Lecture 2), so we can write $L_G = \sum_{i=2}^n \lambda_i u_i u_i^\top$ with orthonormal $u_2, \ldots, u_n \perp \mathbf{1}$ and $\lambda_i > 0$. Let $L_G^\dagger = \sum_{i=2}^n \lambda_i^{-1} u_i u_i^\top$. Then $L_G L_G^\dagger \chi_e = \chi_e$ because $\chi_e \perp \mathbf{1}$, so $v = L_G^\dagger \chi_e + c\mathbf{1}$ for some $c \in \mathbb{R}$, and

$$\mathrm{ER}_e = \chi_e^\top L_G^\dagger \chi_e = \mathrm{Tr}(L_G^\dagger H_e).$$

Summing over the edges and using $L_G = \sum_e w(e) H_e$,

$$\sum_{e \in E} w(e)\,\mathrm{ER}_e = \mathrm{Tr}\Big(L_G^\dagger \sum_{e \in E} w(e) H_e\Big) = \mathrm{Tr}(L_G^\dagger L_G) = \mathrm{Tr}\Big(\sum_{i=2}^n u_i u_i^\top\Big) = n - 1. \qquad \square$$

By Foster's theorem, the numbers $p_e = w(e)\,\mathrm{ER}_e\,/(n-1)$ form a probability distribution on E. Our random $n \times n$ SPSD matrix Y is defined as follows.

1) Pick e with probability $p_e = \dfrac{w(e)\,\mathrm{ER}_e}{n-1}$.

2) Return $\dfrac{w(e)}{p_e} H_e = \dfrac{(n-1)\,w(e)}{w(e)\,\mathrm{ER}_e} H_e = \dfrac{n-1}{\mathrm{ER}_e} H_e$.

Then

$$\mathbb{E}[Y] = \sum_{e \in E} p_e \Big(\frac{w(e)}{p_e} \Big) H_e = \sum_{e \in E} w(e) H_e = L_G.$$

We need μ such that $\frac{n-1}{\mathrm{ER}_e} H_e \preceq \mu L_G$ for every e, that is,

$$H_e \preceq \mu \Big(\frac{\mathrm{ER}_e}{n-1} \Big) L_G.$$

By Homework 8.11, this holds if and only if $\mathrm{ER}_e \le \mu \, \mathrm{ER}_e / (n-1)$, that is, since $\mathrm{ER}_e > 0$, if and only if $\mu \ge n-1$. Thus $\mu = n$ works, and by the computation of Example 8.9, AW gives concentration with probability at least $1 - 2/n$ once $k \ge c' n \log n$ with $c' = 2/(c\varepsilon^2) = 8/\varepsilon^2$.

The average $\frac{1}{k} \sum_i Y_i$ is the Laplacian of a graph with at most k edges, since each Y_i is a positive multiple of some H_e.

Theorem 8.15 (Spielman and Srivastava [SS11])**.** *Let G be a connected graph on $n \ge 2$ vertices and $0 < \varepsilon \le 1$. Let H be the output of k-Sample with the probabilities $p_e = w(e) \, \mathrm{ER}_e / (n-1)$ and $k \ge 8n \log n / \varepsilon^2$ samples. Then H has at most k edges, and with probability at least $1 - 2/n$,*

$$(1-\varepsilon) L_G \preceq L_H \preceq (1+\varepsilon) L_G.$$

Proof. The Laplacian of H is $\frac{1}{k} \sum_{i=1}^{k} Y_i$ for independent copies $Y_1, \dots, Y_k$ of the matrix Y above. Apply AW with $Z = L_G$ and $\mu = n$, and the computation of Example 8.9. □

In particular, for fixed ε, sampling $O(n \log n)$ times in proportion to $w(e) \, \mathrm{ER}_e$ gives, with high probability, a spectral sparsifier with $O(n \log n)$ edges.

Five questions remain at this point; we list them with their answers.

1) Spectral sparsifiers with a linear number of edges exist [BSS12].

2) A spectral sparsifier with $O(n \log n)$ edges can be found quickly.

3) AW still needs a proof; Lecture 11 returns to it.

4) Exact effective resistances are not needed; see the next subsection.

5) Foster's theorem is proved after its statement, Theorem 8.14.

8.6 Upper estimates of effective resistances

Suppose that we know numbers P_e with $P_e \ge \mathrm{ER}_e$ for every $e \in E$. Set

$$t = \sum_{e \in E} w_e P_e,$$

so that $t \ge \sum_e w_e \, \mathrm{ER}_e = n - 1$ by Foster's theorem. Our probabilities are $P'_e = w_e P_e / t$, and Y is defined as follows.

1) Pick e with probability P'_e.

2) Return $\dfrac{w_e}{P'_e} H_e = \dfrac{t}{P_e} H_e$.

As before, $\mathbb{E}[Y] = \sum_e P'_e(w_e/P'_e)H_e = L_G$. Further, $Y \preceq \mu L_G$ for every outcome if $\frac{t}{P_e}H_e \preceq \mu L_G$ for every e, and by Homework 8.11,

$$\begin{aligned} \frac{t}{P_e}H_e \preceq \mu L_G \quad &\text{if and only if} \quad H_e \preceq \frac{P_e}{t}\mu L_G \\ &\text{if and only if} \quad \mathrm{ER}_e \le \frac{P_e}{t}\mu \\ &\text{if and only if} \quad \mu \ge t\frac{\mathrm{ER}_e}{P_e}. \end{aligned}$$

Since $\mathrm{ER}_e \le P_e$, the last condition holds for every e whenever $\mu \ge t$. With $\mu = t$, the computation of Example 8.9 shows that AW gives concentration for $k \ge 8t \log n/\varepsilon^2$, which is $O(t \log t)$ for fixed ε because $n \le t + 1$. Since $t \ge n - 1$, using the estimates P_e may need more samples.

9 Bounding Eigenvalues, Courant–Fischer, Path Embedding

Lecture of February 5, 2020. Based on Gary L. Miller's handwritten notes. Lichen Zhang's scribe notes for this lecture were used as a reading aid and for a few explanatory sentences.

This lecture develops tools for bounding the eigenvalues of graph matrices. The central tool is the Courant–Fischer theorem, which describes every eigenvalue of a symmetric matrix as the value of an optimization problem over subspaces. We use it to bound the largest adjacency eigenvalue and the largest Laplacian eigenvalue in terms of degrees, to show that the Loewner order on matrices orders their eigenvalues, and to bound λ_2 of the path P_n from above by a test vector. For a matching lower bound we introduce path embeddings, which compare the Laplacians of two graphs, and embed the complete graph K_n into P_n. The conclusion is that $\lambda_2(P_n)$ lies between $4/n^2$ and $12/n^2$.

9.1 Mediant of fractions

Definition 9.1. Let $a_1, \dots, a_n$ be real numbers and $b_1, \dots, b_n > 0$. The *mediant* of the fractions $a_1/b_1, \dots, a_n/b_n$ is

$$\frac{\sum_{i=1}^n a_i}{\sum_{i=1}^n b_i}.$$

Lemma 9.2 (Mediant inequality)**.** *Let $a_1, \dots, a_n \in \mathbb{R}$ and $b_1, \dots, b_n > 0$. Then*

$$\min_i \frac{a_i}{b_i} \le \frac{\sum_i a_i}{\sum_i b_i} \le \max_i \frac{a_i}{b_i}.$$

Proof. Consider the points $P_i = (b_i, a_i)$ in the right half-plane, so that the slope of the line from the origin to P_i is $\mathrm{Slope}(P_i) = a_i/b_i$. Their center of mass is $P = \frac{1}{n}\big(\sum_i b_i, \sum_i a_i\big)$, and its slope is the mediant. The slope of P is a weighted average of the slopes of the P_i:

$$\mathrm{Slope}(P) = \frac{\sum_i a_i}{\sum_j b_j} = \sum_i \frac{b_i}{\sum_j b_j} \cdot \frac{a_i}{b_i},$$

with positive weights $b_i / \sum_j b_j$ that sum to 1. A weighted average is at most the largest of the averaged numbers, and at least the smallest. □

9.2 Courant–Fischer (CF)

Throughout this lecture, the eigenvalues of a symmetric matrix A that is not a Laplacian are listed in decreasing order, $\mu_1 \ge \mu_2 \ge \cdots \ge \mu_n$. For $x \neq 0$, the *Rayleigh quotient* of A at x is $x^\top Ax / x^\top x$. All minima and maxima of Rayleigh quotients over a subspace are taken over its nonzero vectors.

Theorem 9.3 (Courant–Fischer)**.** *Let $A \in \mathbb{R}^{n \times n}$ be symmetric with eigenvalues $\mu_1 \ge \cdots \ge \mu_n$. Then for every $k \in \{1, \dots, n\}$,*

$$\mu_k = \max_{\substack{S \subseteq \mathbb{R}^n \\ \dim(S) = k}} \min_{x \in S} \frac{x^\top Ax}{x^\top x} = \min_{\substack{T \subseteq \mathbb{R}^n \\ \dim(T) = n-k+1}} \max_{x \in T} \frac{x^\top Ax}{x^\top x}, \tag{9.1}$$

where S and T range over linear subspaces.

Remark 9.4. We can view the max–min form as a two-person game. The first player chooses a subspace S of dimension k, and the second player then chooses a nonzero vector $x \in S$ to make the Rayleigh quotient as small as possible. The theorem says that μ_k is the value of this game.

By the spectral theorem, A has orthonormal eigenvectors $x_1, \dots, x_n$ with $Ax_i = \mu_i x_i$. If $x = \sum_i c_i x_i$, then $x^\top x = \sum_i c_i^2$ and $x^\top A x = \sum_i \mu_i c_i^2$.

Proof. We first prove the max–min form, in two steps.

($\geq$) Take $S = \langle x_1, \dots, x_k \rangle$ and a nonzero $x \in S$, say $x = \sum_{i=1}^{k} c_i x_i$. Since $\mu_i \geq \mu_k$ for every $i \leq k$, the mediant inequality (Lemma 9.2, applied to the indices with $c_i \neq 0$, where the fractions are $\mu_i c_i^2 / c_i^2 = \mu_i$) gives

$$\frac{x^\top A x}{x^\top x} = \frac{\sum_{i \leq k} \mu_i c_i^2}{\sum_{i \leq k} c_i^2} \geq \mu_k.$$

Hence the minimum over this particular S is at least μ_k, and so is the maximum over all S.

($\leq$) We show that every subspace S with $\dim(S) = k$ satisfies

$$\min_{x \in S} \frac{x^\top A x}{x^\top x} \leq \mu_k.$$

Let $T = \langle x_k, \dots, x_n \rangle$, so that $\dim(T) = n - k + 1$. Since $\dim(S) + \dim(T) = n + 1 > n$, the intersection $S \cap T$ contains a nonzero vector x, and the minimum over S is at most the value at this x. Write $x = \sum_{i=k}^{n} c_i x_i$. Since $\mu_i \leq \mu_k$ for $i \geq k$, the mediant inequality, again applied to the indices with $c_i \neq 0$, gives

$$\frac{x^\top A x}{x^\top x} = \frac{\sum_{k \leq i \leq n} \mu_i c_i^2}{\sum_{k \leq i \leq n} c_i^2} \leq \mu_k.$$

Since S was arbitrary, the maximum over all S is at most μ_k.

The min–max form follows by applying the max–min form to $-A$. The eigenvalues of $-A$ in decreasing order are $-\mu_n \geq \cdots \geq -\mu_1$, so $-\mu_k$ is its $(n-k+1)$-th largest eigenvalue, and

$$-\mu_k = \max_{\dim(T) = n-k+1} \min_{x \in T} \Big(-\frac{x^\top A x}{x^\top x} \Big) = - \min_{\dim(T) = n-k+1} \max_{x \in T} \frac{x^\top A x}{x^\top x}. \qquad \square$$

9.3 Degrees and the extreme eigenvalues

Let $G = (V, E, w)$ be a graph on n vertices with nonnegative weights, adjacency matrix A and degree matrix D. Let $\mu_1 \geq \cdots \geq \mu_n$ be the eigenvalues of A, and let $d_{\max}$ and $d_{\text{avg}} = \frac{1}{n} \sum_v d_v$ be the maximum and the average degree. For unit weights, $d_{\text{avg}} = 2m/n$.

Lemma 9.5.

$$d_{\text{avg}} \overset{(a)}{\leq} \mu_1 \overset{(b)}{\leq} d_{\max}.$$

Proof. (a) By Courant–Fischer with $k = 1$, and with the test vector $\mathbf{1}$,

$$\mu_1 = \max_{x \neq 0} \frac{x^\top A x}{x^\top x} \geq \frac{\mathbf{1}^\top A \mathbf{1}}{\mathbf{1}^\top \mathbf{1}} = \frac{\sum_v d_v}{n} = d_{\text{avg}}.$$

(b) Let $Ax = \mu_1 x$ with $x \neq 0$. Replacing x by $-x$ if necessary, the largest entry of x is positive; let v be a vertex with $x_v \geq x_u$ for all $u \in V$, so $x_v > 0$. Since the entries of A are nonnegative,

$$\mu_1 = \frac{(Ax)_v}{x_v} = \frac{\sum_u A_{vu} x_u}{x_v} \leq \sum_u A_{vu} = d_v \leq d_{\max}. \qquad \square$$

For the next lemma, G has unit weights.

Lemma 9.6. *Let G be a connected graph with unit weights. Then G is d-regular if and only if $\mu_1 = d = d_{\max}$.*

Proof. ($\Rightarrow$) If G is d-regular, then $d_{\max} = d$, and

$$A \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} = d \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}.$$

Moreover $d_{\text{avg}} = d_{\max} = d$, so $\mu_1 = d$ by Lemma 9.5.

($\Leftarrow$) Suppose $\mu_1 = d = d_{\max}$, and let $Ax = dx$ with $x \neq 0$. As in the proof of Lemma 9.5, we may assume that the largest entry x_v of x is positive. Then

$$dx_v = \sum_{(v,u)\in E} x_u,$$

and we claim that $x_u = x_v$ for every neighbor u of v. Indeed, each $x_u \leq x_v$, and v has $\deg(v) \leq d$ neighbors, so

$$dx_v = \sum_{(v,u)\in E} x_u \leq \deg(v)\, x_v \leq dx_v.$$

Equality holds throughout, which forces $x_u = x_v$ for every neighbor u (and $\deg(v) = d$). Every neighbor of v therefore also attains the largest entry, and the same argument applies to it. Since G is connected, $x_u = x_v$ for all $u \in V$. Hence x is a multiple of $\mathbf{1}$, and $A\mathbf{1} = d\mathbf{1}$ says that every vertex has degree d. □

Lemma 9.7. *Let G be a graph with nonnegative weights and Laplacian $L = L(G)$, with eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. Then $\lambda_n \leq 2d_{\max}$.*

Proof. Since $L = D - A$, Courant–Fischer gives

$$\lambda_n = \max_{x\neq 0} \frac{x^\top Dx - x^\top Ax}{x^\top x} \leq \lambda_{\max}(D) - \lambda_{\min}(A) \leq d_{\max} - (-d_{\max}) = 2d_{\max}.$$

Here $\lambda_{\max}(D) = d_{\max}$ because D is diagonal. For the bound $\lambda_{\min}(A) \geq -d_{\max}$, let $|x|$ denote the vector with entries $|x_i|$. Since A has nonnegative entries, $|x^\top Ax| \leq |x|^\top A|x| \leq \mu_1\, x^\top x$, so $\lambda_{\min}(A) \geq -\mu_1 \geq -d_{\max}$ by Courant–Fischer and Lemma 9.5. □

9.4 The Loewner order

Definition 9.8. Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric. We write $A \succeq 0$ if $x^\top Ax \geq 0$ for all $x \in \mathbb{R}^n$. More generally, $A \preceq B$ if $x^\top Ax \leq x^\top Bx$ for all x, that is, if $B - A \succeq 0$. This partial order is the *Loewner order*.

The order is transitive: if $A \preceq B$ and $B \preceq C$, then $A \preceq C$. It is compatible with addition: if $A \preceq B$, then $A + C \preceq B + C$ for every symmetric C. Both facts follow at once from the definition.

For graphs G and H on the same vertex set, we write $G \preceq H$ if $L_G \preceq L_H$. If $G \cup H$ denotes the graph on the same vertex set whose edge weights are the sums of those of G and H, then $L_{G\cup H} = L_G + L_H$ and $L_H \succeq 0$, so $G \preceq G \cup H$.

Theorem 9.9. *Let $A, B \in \mathbb{R}^{n\times n}$ be symmetric with $A \preceq B$, and let $\lambda_k(\cdot)$ denote the k-th largest eigenvalue. Then $\lambda_k(A) \le \lambda_k(B)$ for every $k \in \{1, \dots, n\}$.*

Proof. For every subspace T and every nonzero $x \in T$, the Rayleigh quotient of B at x is at least that of A. Hence, by Courant–Fischer,

$$\lambda_k(B) = \min_{\dim(T)=n-k+1} \max_{x\in T} \frac{x^\top B x}{x^\top x} \ge \min_{\dim(T)=n-k+1} \max_{x\in T} \frac{x^\top A x}{x^\top x} = \lambda_k(A). \qquad \square$$

Since the k-th smallest eigenvalue is the $(n-k+1)$-th largest, the theorem holds equally for eigenvalues listed in increasing order. This is how we use it for λ_2 of Laplacians below.

9.5 Pseudo-inverse

Definition 9.10. For $a \in \mathbb{R}$, let

$$a^\dagger = \begin{cases} 0 & \text{if } a = 0, \\ a^{-1} & \text{otherwise.} \end{cases}$$

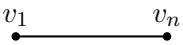
For a diagonal matrix D, let $D^\dagger$ be the diagonal matrix with $(D^\dagger)_{ii} = D_{ii}^\dagger$.

A symmetric matrix A can be written as $A = U^\top \Lambda U$ with U orthogonal and Λ diagonal, by the spectral theorem.

Definition 9.11. The *pseudo-inverse* of a symmetric matrix $A = U^\top \Lambda U$ is $A^\dagger = U^\top \Lambda^\dagger U$.

9.6 Effective resistance and the Loewner order

Let $G = (V, E, w)$ be a connected graph with positive weights on the vertex set $V = \{v_1, \dots, v_n\}$, and let ER_{ij} denote the effective resistance between v_i and v_j in G (Lectures 2 and 3). Let $G_{1n} = (V, \{(v_1, v_n)\})$ be the graph on the same vertex set whose only edge, of weight 1, joins v_1 and v_n:



Its Laplacian $L_{G_{1n}}$ satisfies $x^\top L_{G_{1n}} x = (x_1 - x_n)^2$.

Proposition 9.12. *For every real α, we have $G_{1n} \preceq \alpha G$ if and only if $\mathrm{ER}_{1n} \le \alpha$.*

We prove that $G_{1n} \preceq \alpha G$ implies $\mathrm{ER}_{1n} \le \alpha$; the converse is Homework 9.13.

Proof of the forward direction. Let f be the unit potential flow (the electrical flow) from v_1 to v_n, and let φ be its vector of potentials. Its energy is $f^\top R f = \varphi^\top L \varphi$ (Lecture 3), and it equals ER_{1n}. A unit flow has potential drop $\varphi_1 - \varphi_n = \mathrm{ER}_{1n}$, so $\varphi^\top L_{G_{1n}} \varphi = (\mathrm{ER}_{1n})^2$. Applying $G_{1n} \preceq \alpha G$ to the vector φ gives

$$(\mathrm{ER}_{1n})^2 = \varphi^\top L_{G_{1n}} \varphi \le \alpha\, \varphi^\top L \varphi = \alpha\, \mathrm{ER}_{1n}\,.$$

Since $\mathrm{ER}_{1n} > 0$, dividing by ER_{1n} gives $\mathrm{ER}_{1n} \le \alpha$. $\square$

Homework 9.13. Show that $\mathrm{ER}_{1n} \le \alpha$ implies $G_{1n} \preceq \alpha G$.

9.7 Upper bounding λ_2

Let L be the Laplacian of a graph, with eigenvalues $0 = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$. Since $L\mathbf{1} = 0$, we may choose an orthonormal eigenbasis $u_1, \dots, u_n$ with $u_1 = \mathbf{1}/\sqrt{n}$ and $Lu_i = \lambda_i u_i$. A nonzero $x \perp \mathbf{1}$ has the form $x = \sum_{i \ge 2} c_i u_i$, so, as in the proof of Courant–Fischer, $x^\top Lx / x^\top x = \sum_{i \ge 2} \lambda_i c_i^2 / \sum_{i \ge 2} c_i^2 \ge \lambda_2$, with equality at $x = u_2$. Hence

$$\lambda_2 = \min_{\substack{v \ne 0 \\ v^\top \mathbf{1} = 0}} \frac{v^\top L v}{v^\top v}.$$

Thus every nonzero x with $x \perp \mathbf{1}$ satisfies

$$\lambda_2 \le \frac{x^\top L x}{x^\top x}. \tag{9.2}$$

To bound λ_2 from above, it suffices to exhibit one good test vector.

Example 9.14. Let P_n be the path on the vertices $1, \dots, n$ with unit weights, and take the test vector with evenly spaced entries

$$x = \Big(-\frac{n-1}{2}, -\frac{n-1}{2} + 1, \dots, \frac{n-1}{2} - 1, \frac{n-1}{2}\Big),$$

that is, $x_i = i - \frac{n+1}{2}$. The entries are symmetric about 0, so $\mathbf{1}^\top x = 0$. Consecutive entries differ by 1, and

$$x^\top L_{P_n} x = \sum_{i=1}^{n-1} (x_i - x_{i+1})^2 = n - 1,$$
$$x^\top x = \sum_{i=1}^{n} \Big(i - \frac{n+1}{2}\Big)^2 = \frac{n(n^2-1)}{12}.$$

Thus, by (9.2),

$$\lambda_2(P_n) \le \frac{n-1}{n(n^2-1)/12} = \frac{12}{n(n+1)} \le \frac{12}{n^2}.$$

To see that this bound is tight up to a constant factor, we need a matching lower bound on $\lambda_2(P_n)$. We obtain one by comparing P_n with the complete graph, whose spectrum we compute first.

9.8 Lower bounding λ_2

Proposition 9.15. *The Laplacian of the complete graph K_n with unit weights has eigenvalues 0 (once) and n (with multiplicity $n-1$).*

Proof. The Laplacian is

$$L(K_n) = \begin{pmatrix} n-1 & -1 & \cdots & -1 \\ -1 & n-1 & \cdots & -1 \\ \vdots & & \ddots & \vdots \\ -1 & -1 & \cdots & n-1 \end{pmatrix} = nI - \mathbf{1}\mathbf{1}^\top.$$

For example,

$$L(K_n)\begin{pmatrix}1\\-1\\0\\\vdots\\0\end{pmatrix} = \begin{pmatrix}n\\-n\\0\\\vdots\\0\end{pmatrix} = n\begin{pmatrix}1\\-1\\0\\\vdots\\0\end{pmatrix}.$$

In the same way, since $\mathbf{1}^\top(e_i - e_{i+1}) = 0$, each of the vectors

$$x_1 = \begin{pmatrix}1\\-1\\0\\\vdots\\0\end{pmatrix}, \quad x_2 = \begin{pmatrix}0\\1\\-1\\0\\\vdots\end{pmatrix}, \quad \ldots, \quad x_{n-1} = \begin{pmatrix}0\\\vdots\\0\\1\\-1\end{pmatrix}$$

satisfies $L(K_n)x_i = nx_i$. These $n-1$ vectors are linearly independent, so n has multiplicity at least $n-1$. The remaining eigenvalue is 0, with eigenvector $\mathbf{1}$. □

9.9 Path embedding

Definition 9.16. Let $G = (V, E_G)$ (the *guest*) and $H = (V, E_H)$ (the *host*) be graphs with unit weights on the same vertex set. A *path embedding* of G into H is a map

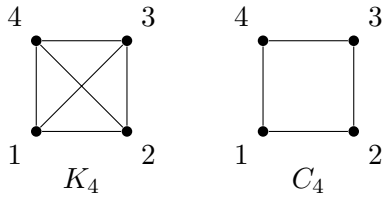
$$f\colon E_G \to \{\text{paths in } H\}$$

such that, for every edge $(u, v) \in E_G$, $f((u, v))$ is a path from u to v in H.

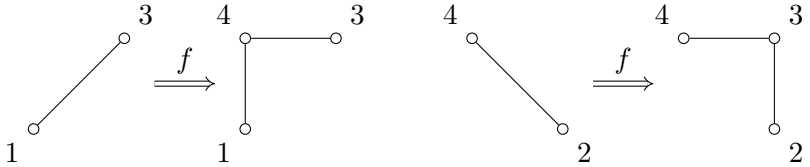
Definition 9.17. The *congestion* of a path embedding f is the largest number of paths $f(e)$, $e \in E_G$, that use a single edge of H, and its *dilation* is the length (number of edges) of the longest path:

$$\text{congestion} = \max_{e' \in E_H} |\{e \in E_G \mid e' \in f(e)\}|, \qquad \text{dilation} = \max_{e \in E_G} |f(e)|.$$

Example 9.18. Let $G = K_4$ and $H = C_4$ on the vertices $1, 2, 3, 4$:



Each edge of K_4 that is also an edge of C_4 is mapped to itself. The two diagonals are mapped to paths of length 2: $f((1, 3))$ is the path $1, 4, 3$, and $f((2, 4))$ is the path $2, 3, 4$:



The edge $(3, 4)$ of C_4 lies on three paths (its own and both diagonal paths), and every other edge lies on at most two. Hence the congestion is 3, and the dilation is 2.

9.10 Laplacian as a sum of Laplacians

For $i \neq j$, let E_{ij} and P_{ij} be the $n \times n$ matrices

$$E_{ij} = \begin{pmatrix} \ddots & & & \\ & 1 & -1 & \\ & -1 & 1 & \\ & & & \ddots \end{pmatrix}, \qquad P_{ij} = \begin{pmatrix} \ddots & & & \\ & 1 & 1 & \\ & 1 & 1 & \\ & & & \ddots \end{pmatrix},$$

whose only nonzero entries lie in rows and columns i and j. Thus $E_{ij} = (e_i - e_j)(e_i - e_j)^\top$ is the Laplacian of the single edge (i, j), and $P_{ij} = (e_i + e_j)(e_i + e_j)^\top$. Both are positive semidefinite.

Proposition 9.19. *The Laplacian of $G = (V, E, w)$ is $L_G = \sum_{(i,j)\in E} w_{ij} E_{ij}$, with each edge counted once.*

Proof. Write $L_G = BCB^\top$ (Lecture 2) as a sum of outer products of the columns of B. The column of an edge (i, j) is $\pm(e_i - e_j)$, and its outer product with itself is E_{ij}. □

Definition 9.20. A matrix $A \in \mathbb{R}^{n\times n}$ is *diagonally dominant* if $A_{ii} \geq \sum_{j\neq i} |A_{ij}|$ for every i. It is *SDD* (symmetric diagonally dominant) if it is also symmetric.

Proposition 9.21. *If A is SDD, then*

$$A = \sum_{\substack{i<j \\ A_{ij}<0}} (-A_{ij})\, E_{ij} + \sum_{\substack{i<j \\ A_{ij}>0}} A_{ij} P_{ij} + D,$$

where D is diagonal with nonnegative entries.

Proof. Let A' denote the sum of the first two terms. For $i < j$, the (i, j) entry of A' comes only from the term of the pair $\{i, j\}$, and equals A_{ij} in both cases. The (i, i) entry of A' is $\sum_{j\neq i} |A_{ij}|$, since each pair $\{i, j\}$ with $A_{ij} \neq 0$ contributes $|A_{ij}|$. Hence $D = A - A'$ is diagonal with $D_{ii} = A_{ii} - \sum_{j\neq i} |A_{ij}| \geq 0$. □

Theorem 9.22. *Let G and H be graphs with unit weights on the same vertex set. If $f : G \to H$ is a path embedding with congestion c and dilation d, then*

$$L_G \preceq c \cdot d\, L_H.$$

Proof. Let $e_1, \ldots, e_m$ be the edges of G, and let $E_1, \ldots, E_m$ be their edge Laplacians, so that $L_G = \sum_i E_i$ by Proposition 9.19. For each i, let P_i be the *path Laplacian* of $f(e_i)$, the sum of the edge Laplacians of the edges of H on the path $f(e_i)$.

First, $\sum_i P_i \preceq c\, L_H$. Indeed, $\sum_i P_i = \sum_{e'\in E_H} n_{e'} L_{e'}$, where $L_{e'}$ is the edge Laplacian of e' and $n_{e'}$ is the number of paths $f(e_i)$ that use e'. By the definition of congestion $n_{e'} \leq c$, and $L_{e'} \succeq 0$, so $\sum_i P_i \preceq c \sum_{e'} L_{e'} = c\, L_H$.

Next, fix $x \in \mathbb{R}^n$. For each i, let $e_i = (u_0, u_\ell)$ and let $f(e_i)$ be the path $u_0, u_1, \ldots, u_\ell$, with $\ell \leq d$. By the Cauchy–Schwarz inequality,

$$x^\top E_i x = \Big(\sum_{t=1}^{\ell} (x_{u_{t-1}} - x_{u_t})\Big)^2 \leq \ell \sum_{t=1}^{\ell} (x_{u_{t-1}} - x_{u_t})^2 \leq d\, x^\top P_i x.$$

In particular, $x^\top E_i x = 0$ whenever $x^\top P_i x = 0$. Let I be the set of indices i with $x^\top P_i x > 0$. If I is empty, then $x^\top L_G x = 0 \le c\,d\,x^\top L_H x$. Otherwise, the mediant inequality gives

$$\frac{x^\top L_G x}{\sum_i x^\top P_i x} = \frac{\sum_{i\in I} x^\top E_i x}{\sum_{i\in I} x^\top P_i x} \le \max_{i\in I} \frac{x^\top E_i x}{x^\top P_i x} \le d.$$

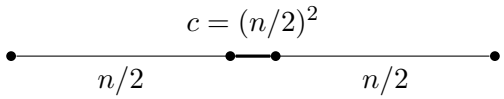
Combining this with the first step,

$$x^\top L_G x \le d \sum_i x^\top P_i x \le c \cdot d\, x^\top L_H x.$$

Since x was arbitrary, $L_G \preceq c\,d\,L_H$. □

9.11 Back to λ_2 for P_n

Let $f\colon K_n \to P_n$ be the path embedding that maps each edge (i, j) of K_n, with $i < j$, to the path $i, i+1, \dots, j$; since P_n is a tree, this is the only path embedding. The edge $(k, k+1)$ of P_n lies on the paths of the $k(n-k)$ pairs with $i \le k < j$. This number is largest at the middle edge:



so the congestion is $c = \lfloor n/2 \rfloor \lceil n/2 \rceil \le (n/2)^2$, with equality for even n. The dilation is $d = n - 1$, from the edge $(1, n)$.

Theorem 9.23. *For every $n \ge 2$,*

$$\frac{4}{n^2} \le \lambda_2(P_n) \le \frac{12}{n^2}.$$

Proof. The upper bound is Example 9.14. For the lower bound, the embedding theorem (Theorem 9.22) gives $L_{K_n} \preceq c\,d\,L_{P_n}$, and Theorem 9.9, applied to the second-smallest eigenvalue, gives $\lambda_2(K_n) \le c\,d\,\lambda_2(P_n)$. Since $\lambda_2(K_n) = n$ by Proposition 9.15,

$$\lambda_2(P_n) \ge \frac{n}{c \cdot d} \ge \frac{n}{(n/2)^2(n-1)} = \frac{4}{n(n-1)} \ge \frac{4}{n^2}.$$

□

10 Golden–Thompson Inequality via Weyl's Majorant Theorem

Lecture of February 7, 2020. Based on Gary L. Miller's handwritten notes.

For real numbers, $e^{a+b} = e^a e^b$. For matrices this identity fails, and the Golden–Thompson inequality is what survives of it under the trace: $\operatorname{Tr}(e^{A+B}) \le \operatorname{Tr}(e^A e^B)$ for symmetric A and B. This lecture proves it. The proof uses both eigenvalues and singular values, so we first collect the necessary linear algebra over the complex numbers: the Schur decomposition, the singular value decomposition and the polar decomposition. The key tools are Weyl's majorant theorem, which bounds products of the largest eigenvalues (in modulus) by products of the largest singular values, and the Lie product formula, which writes e^{A+B} as a limit of products of $e^{A/N}$ and $e^{B/N}$. We prove Weyl's theorem with a Courant–Fischer theorem for singular values (Lecture 9) and an interlacing theorem.

10.1 Golden–Thompson inequality

Theorem 10.1 (Golden–Thompson)**.** *For all symmetric $A, B \in \mathbb{R}^{n\times n}$,*

$$\operatorname{Tr}(e^{A+B}) \le \operatorname{Tr}(e^A e^B). \tag{10.1}$$

Remark 10.2. For symmetric A and B, $e^{A+B} = e^A e^B$ if and only if $AB = BA$; we do not use this fact and omit its proof. The inequality does not extend to three matrices: in general,

$$\operatorname{Tr}(e^{A+B+C}) \not\le \operatorname{Tr}(e^A e^B e^C).$$

The proof uses both eigenvalues and singular values. For intuition, it helps to think of e^A as a transition matrix.

Definition 10.3. A matrix B is *similar* to A if there is an invertible P such that $B = P^{-1}AP$.

Similarity preserves the rank, the eigenvalues, the trace and the minimal polynomial. It does not preserve the angles between eigenvectors, nor the singular values (defined below).

10.2 Real linear algebra needs complex numbers

A real matrix can have nonreal eigenvalues, so even for real matrices we need complex vectors and complex matrices. Let $\mathbb{C} = \{a + ib \mid a, b \in \mathbb{R}\}$, and let $z^* = (a+ib)^* = a - ib$ denote the complex conjugate.

Definition 10.4. The *conjugate transpose* (or *Hermitian transpose*) of $A \in \mathbb{C}^{p\times q}$ is the matrix $A^* \in \mathbb{C}^{q\times p}$ with $(A^*)_{ij} = A^*_{ji}$. A matrix $U \in \mathbb{C}^{n\times n}$ is *unitary* if $U^*U = I$.

If U is unitary, then $U^* = U^{-1}$, and hence also $UU^* = I$. Thus the unitary matrices are closed under conjugate transposes and under products. On $\mathbb{C}^n$ we use the inner product $\langle a, b\rangle = a^*b$. Unitary matrices preserve it: $\langle Ua, Ub\rangle = a^*U^*Ub = \langle a, b\rangle$.

Definition 10.5. A matrix A is *unitarily similar* to B if $A = U^{-1}BU = U^*BU$ for some unitary U.

Theorem 10.6 (Schur decomposition)**.** *Let $A \in \mathbb{C}^{n\times n}$ have eigenvalues $\lambda_1, \dots, \lambda_n$ (with multiplicity), ordered so that $|\lambda_1| \ge |\lambda_2| \ge \cdots \ge |\lambda_n|$. Then there is a unitary U such that U^*AU is upper triangular with $(U^*AU)_{ii} = \lambda_i$ for every i.*

Proof. The proof is Gram–Schmidt orthogonalization, applied repeatedly. We induct on n; the case $n = 1$ is trivial. Let u_1 be a unit eigenvector for λ_1. By Gram–Schmidt, extend u_1 to an orthonormal basis $u_1, \dots, u_n$ of $\mathbb{C}^n$, and let U_1 be the unitary matrix with columns $u_1, \dots, u_n$. The first column of $U_1^* A U_1$ is $U_1^* A u_1 = \lambda_1 U_1^* u_1 = \lambda_1 e_1$, so

$$U_1^* A U_1 = \begin{pmatrix} \lambda_1 & w^* \\ 0 & A' \end{pmatrix}$$

for some $w \in \mathbb{C}^{n-1}$ and $A' \in \mathbb{C}^{(n-1)\times(n-1)}$. Expanding the characteristic polynomial along the first column shows that the eigenvalues of A' are $\lambda_2, \dots, \lambda_n$. By induction there is a unitary U' such that $U'^* A' U'$ is upper triangular with diagonal $\lambda_2, \dots, \lambda_n$. Then $U = U_1 \left(\begin{smallmatrix} 1 & 0 \\ 0 & U' \end{smallmatrix}\right)$ is unitary, and $U^* A U$ is upper triangular with diagonal $\lambda_1, \dots, \lambda_n$. □

Thus a matrix in $\mathbb{C}^{n\times n}$ always has n eigenvalues, counted with multiplicity, although it may not have n linearly independent eigenvectors.

Theorem 10.7 (Singular value decomposition)**.** *Let $A \in \mathbb{R}^{n\times n}$ be nonzero, and let $r = \operatorname{rank}(A)$. There are $U, V \in \mathbb{R}^{n\times r}$ with $U^\top U = V^\top V = I$ and*

$$A = U \Lambda V^\top, \qquad \text{where } \Lambda = \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{pmatrix}, \quad \sigma_1 \geq \cdots \geq \sigma_r > 0.$$

Proof. The matrix $A^\top A$ is SPSD (symmetric positive semidefinite), since $x^\top A^\top A x = \|Ax\|_2^2 \geq 0$. By the spectral theorem, $A^\top A = \bar{Q} \bar{\Lambda}^2 \bar{Q}^\top$ with $\bar{Q}$ orthogonal and $\bar{\Lambda}$ diagonal with nonnegative entries, in decreasing order. If $A^\top A x = 0$, then $\|Ax\|_2^2 = x^\top A^\top A x = 0$, so $\operatorname{Ker}(A^\top A) = \operatorname{Ker}(A)$, and $A^\top A$ has exactly r positive eigenvalues. Let $Q \in \mathbb{R}^{n\times r}$ be $\bar{Q}$ with the eigenvectors of eigenvalue zero removed, and let Λ be $\bar{\Lambda}$ with the zero entries removed. Thus

$$A^\top A = Q \Lambda^2 Q^\top, \qquad Q^\top Q = I.$$

We take $V = Q$, and it remains to find U. Note that

$$(AA^\top) AQ = A(A^\top A Q) = AQ\Lambda^2,$$

so the columns of AQ are eigenvectors of $AA^\top$ with eigenvalues $\sigma_1^2, \dots, \sigma_r^2$. Set $U = AQ\Lambda^{-1}$. Then

$$U^\top U = \Lambda^{-1} Q^\top A^\top A Q \Lambda^{-1} = \Lambda^{-1} Q^\top Q \Lambda^2 Q^\top Q \Lambda^{-1} = I,$$

and

$$U^\top A Q = \Lambda^{-1} Q^\top A^\top A Q = \Lambda^{-1} Q^\top Q \Lambda^2 = \Lambda.$$

Finally, $AQ = U\Lambda$ gives $U\Lambda Q^\top = AQQ^\top$. The matrix $I - QQ^\top$ is the orthogonal projection onto the span of the removed eigenvectors, which is $\operatorname{Ker}(A^\top A) = \operatorname{Ker}(A)$, so $A(I - QQ^\top) = 0$. Thus $A = U\Lambda Q^\top$, and then $AA^\top = U\Lambda^2 U^\top$. □

Definition 10.8. The diagonal entries $\sigma_1, \dots, \sigma_r$ of Λ are the *singular values* of A. More generally, for $A \in \mathbb{C}^{p\times q}$ the matrix A^*A is Hermitian and positive semidefinite, and the singular values $\sigma_1(A) \geq \cdots \geq \sigma_q(A) \geq 0$ are the square roots of its eigenvalues. For real $A \in \mathbb{R}^{n\times n}$ these are $\sigma_1, \dots, \sigma_r$ followed by $n - r$ zeros, because $A^\top A = Q\Lambda^2 Q^\top$. We write $\operatorname{sing}(A)$ for the list of singular values. The proof of Theorem 10.7 works verbatim over $\mathbb{C}$, with A^* in place of $A^\top$.

The singular value decomposition gives a second factorization. Since $V^\top V = I$ and $U^\top U = I$,

$$A = U\Lambda V^\top V U^\top U V^\top = U\Lambda U^\top (UV^\top).$$

When $r < n$, the matrix $UV^\top$ has rank r and is not orthogonal. To repair this, extend the columns of U and of V to orthonormal bases of $\mathbb{R}^n$, giving orthogonal matrices $\hat{U}, \hat{V} \in \mathbb{R}^{n\times n}$, and let $\hat{\Lambda} = \operatorname{diag}(\sigma_1, \dots, \sigma_r, 0, \dots, 0)$. Then still $A = \hat{U}\hat{\Lambda}\hat{V}^\top$, and the same computation gives

$$A = \hat{U}\hat{\Lambda}\hat{U}^\top(\hat{U}\hat{V}^\top) = M \cdot P,$$

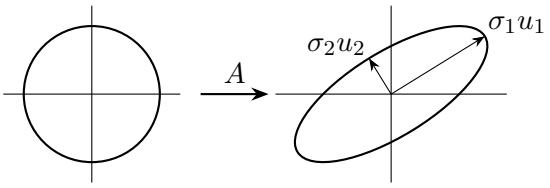
with $M = \hat{U}\hat{\Lambda}\hat{U}^\top$ SPSD and $P = \hat{U}\hat{V}^\top$ orthogonal (that is, real and unitary).

Theorem 10.9 (Polar decomposition)**.** *For every $A \in \mathbb{R}^{n\times n}$ there are an SPSD matrix M and an orthogonal matrix P such that $A = MP$.*

Example 10.10. Let

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Its eigenvalues are $\lambda_1 = \lambda_2 = 1$, and $\sigma_1 \cdot \sigma_2 = |\det(A)| = 1$. Explicitly, $A^\top A = \left(\begin{smallmatrix} 1 & 1 \\ 1 & 2 \end{smallmatrix}\right)$ has eigenvalues $(3 \pm \sqrt{5})/2$, so $\sigma_1 = (1+\sqrt{5})/2$ and $\sigma_2 = (\sqrt{5}-1)/2$. The matrix A maps the unit circle onto an ellipse whose semi-axes are $\sigma_1 u_1$ and $\sigma_2 u_2$, where u_1, u_2 are the left singular vectors (the columns of U):



Weyl's theorem (Theorem 10.12 below) with $k = 1$ gives $|\lambda_1| \le \sigma_1$, here $1 \le (1+\sqrt{5})/2$, and its corollary (Corollary 10.13) with $k = 2$ and $m = 1$ gives $\lambda_1 + \lambda_2 \le \sigma_1 + \sigma_2$, here $2 \le \sqrt{5}$. With $k = 2$, both sides of Weyl's inequality equal $|\det(A)| = 1$, so $\lambda_2 = 1/\lambda_1 \ge 1/\sigma_1 = \sigma_2$. Thus $\lambda_2 \ge \sigma_2$.

Corollary 10.11. *For every $A \in \mathbb{C}^{n\times n}$ with eigenvalues $\lambda_1, \dots, \lambda_n$ and singular values $\sigma_1, \dots, \sigma_n$,*

$$|\det(A)| = \Big|\prod_i \lambda_i\Big| = \prod_i \sigma_i.$$

Proof. By the Schur decomposition, $\det(A) = \det(U^*AU) = \prod_i \lambda_i$, the determinant of an upper triangular matrix. Moreover $|\det(A)|^2 = \det(A^*)\det(A) = \det(A^*A) = \prod_i \sigma_i^2$, since the eigenvalues of A^*A are the σ_i^2. □

10.3 Weyl's majorant theorem

Theorem 10.12 (Weyl)**.** *Let $A \in \mathbb{C}^{n\times n}$ have singular values $\sigma_1 \ge \cdots \ge \sigma_n \ge 0$ and eigenvalues $\lambda_1, \dots, \lambda_n$, ordered so that $|\lambda_1| \ge |\lambda_2| \ge \cdots \ge |\lambda_n|$. Then*

$$\Big|\prod_{i=1}^{k} \lambda_i\Big| \le \prod_{i=1}^{k} \sigma_i \qquad \text{for every } k,\ 1 \le k \le n.$$

Corollary 10.13. *Under the hypotheses of Theorem 10.12,*

$$\sum_{i=1}^{k} |\lambda_i|^m \le \sum_{i=1}^{k} \sigma_i^m \qquad \text{for every } k,\ 1 \le k \le n,\ \text{and every } m \in \{1, 2, \dots\}.$$

The corollary is proved after the theorem. The proof of the theorem needs two tools: a Courant–Fischer theorem for singular values, and an interlacing theorem.

Theorem 10.14 (Courant–Fischer for singular values)**.** *Let $A \in \mathbb{C}^{p\times q}$ and $1 \le k \le q$. Then, with T and S ranging over subspaces of $\mathbb{C}^q$ and x over their nonzero vectors,*

$$\sigma_k(A) = \min_{\dim(T)=q-k+1} \max_{x\in T} \frac{\|Ax\|_2}{\|x\|_2},$$
$$\sigma_k(A) = \max_{\dim(S)=k} \min_{x\in S} \frac{\|Ax\|_2}{\|x\|_2}.$$

Proof. We have $\|Ax\|_2^2/\|x\|_2^2 = x^*A^*Ax/x^*x$, the Rayleigh quotient of A^*A, whose k-th largest eigenvalue is $\sigma_k(A)^2$. The Courant–Fischer theorem (Theorem 9.3) and its proof hold verbatim for Hermitian matrices, with x^* in place of $x^\top$. Applying it to A^*A and taking square roots, which preserves the order, gives both formulas. □

Definition 10.15. Let $a_0 \le a_1 \le \cdots \le a_n$ and $b_1 \le \cdots \le b_n$ be real numbers. The sequence $\langle b\rangle$ *interlaces* with $\langle a\rangle$ if

$$a_0 \le b_1 \le a_1 \le b_2 \le \cdots \le b_n \le a_n.$$

Theorem 10.16 (Cauchy)**.** *Let $A \in \mathbb{R}^{(n+1)\times(n+1)}$ be symmetric, and let A_1 be A with row i and column i removed, for some i. Then $\lambda(A_1)$ interlaces with $\lambda(A)$.*

We do not prove Cauchy's theorem; it follows from Courant–Fischer in the same way as the next theorem, which is its analogue for singular values.

Theorem 10.17. *Let $A \in \mathbb{C}^{p\times q}$, and let A_1 be A with a column removed (if $q \ge 2$) or a row removed (if $p \ge 2$). Then the singular values interlace:*

$$\sigma_k(A) \ge \sigma_k(A_1) \ge \sigma_{k+1}(A) \qquad \text{for } 1 \le k \le q-1.$$

If a row is removed, then also $\sigma_q(A) \ge \sigma_q(A_1)$.

Proof. Suppose first that $B = A_1$ is A with the first column removed. Let $W = \{x \in \mathbb{C}^q \mid x_1 = 0\}$. For $y \in \mathbb{C}^{q-1}$, write $\hat{y} = (0, y) \in W$; then $By = A\hat{y}$ and $\|y\|_2 = \|\hat{y}\|_2$, so the quotients of B on a subspace $T' \subseteq \mathbb{C}^{q-1}$ are the quotients of A on the subspace $\hat{T}' = \{\hat{y} \mid y \in T'\}$ of W. Fix $k \le q-1$.

For every subspace $T \subseteq \mathbb{C}^q$ with $\dim T = q-k+1$, the intersection $T \cap W$ has dimension at least $q-k$, so it contains $\hat{T}'$ for some subspace $T' \subseteq \mathbb{C}^{q-1}$ with $\dim T' = q-k = (q-1)-k+1$. Hence

$$\max_{x\in T} \frac{\|Ax\|_2}{\|x\|_2} \ge \max_{x\in T\cap W} \frac{\|Ax\|_2}{\|x\|_2} \ge \max_{y\in T'} \frac{\|By\|_2}{\|y\|_2} \ge \sigma_k(B),$$

by Theorem 10.14 for B. Taking the minimum over T gives $\sigma_k(A) \ge \sigma_k(B)$.

For every subspace $S \subseteq \mathbb{C}^q$ with $\dim S = k+1$, the intersection $S \cap W$ has dimension at least k, so it contains $\hat{S}'$ for some $S' \subseteq \mathbb{C}^{q-1}$ with $\dim S' = k$. Hence

$$\min_{x \in S} \frac{\|Ax\|_2}{\|x\|_2} \leq \min_{x \in S \cap W} \frac{\|Ax\|_2}{\|x\|_2} \leq \min_{y \in S'} \frac{\|By\|_2}{\|y\|_2} \leq \sigma_k(B).$$

Taking the maximum over S gives $\sigma_{k+1}(A) \leq \sigma_k(B)$. Removing another column reduces to this case: replacing A by $A\Pi$ for a permutation matrix Π replaces A^*A by the similar matrix $\Pi^\top A^*A\Pi$, so it does not change the singular values.

Now suppose that A_1 is A with a row a^* removed, so that $A_1^*A_1 = A^*A - aa^*$ and $\|A_1x\|_2^2 = \|Ax\|_2^2 - |a^*x|^2$. Then $\|A_1x\|_2 \leq \|Ax\|_2$ for every x, so $\sigma_k(A_1) \leq \sigma_k(A)$ for all $k \leq q$ by Theorem 10.14. On the subspace $W' = \{x \mid a^*x = 0\}$, of dimension at least $q-1$, the two quotients agree. For S with $\dim S = k+1$, the intersection $S \cap W'$ contains a subspace S' of dimension k, and the second chain above, with W' for W and A_1 for B, gives $\sigma_{k+1}(A) \leq \sigma_k(A_1)$. □

Proof of Weyl's theorem. Let $X \in \mathbb{C}^{n \times n}$. A unitary similarity changes neither the eigenvalues nor the singular values, since $(U^*XU)^*(U^*XU) = U^*X^*XU$. By the Schur decomposition (Theorem 10.6) we may therefore assume that X is upper triangular with

$$\operatorname{diag}(X) = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}, \qquad |\lambda_i| \geq |\lambda_{i+1}|.$$

Let X_k be the upper left $k \times k$ principal submatrix of X. It is upper triangular with diagonal $\lambda_1, \ldots, \lambda_k$. We obtain X_k from X by removing the last $n-k$ columns and then the last $n-k$ rows, one at a time; by Theorem 10.17, no step increases any of the first k singular values. Hence, by Corollary 10.11,

$$|\lambda_1 \cdots \lambda_k| = |\det(X_k)| = \sigma_1(X_k) \cdots \sigma_k(X_k) \leq \sigma_1(X) \cdots \sigma_k(X).$$ □

Proof of Corollary 10.13. Fix k and m, and write $x_i = |\lambda_i|$ and $y_i = \sigma_i$; both sequences are nonincreasing and nonnegative. Let p be the number of indices $i \leq k$ with $x_i > 0$. The terms with $p < i \leq k$ contribute 0 on the left and at least 0 on the right, so it suffices to show $\sum_{i \leq p} x_i^m \leq \sum_{i \leq p} y_i^m$. For $j \leq p$, Weyl's theorem gives $\prod_{i \leq j} y_i \geq \prod_{i \leq j} x_i > 0$, so $y_1, \ldots, y_p > 0$. Set $a_i = \ln x_i$ and $b_i = \ln y_i$ for $i \leq p$. Both sequences are nonincreasing, and $s_j = \sum_{i \leq j}(b_i - a_i) \geq 0$ for $j \leq p$ (with $s_0 = 0$).

Let $g(t) = e^{mt}$, which is increasing and convex. For $i \leq p$, let c_i be the slope of the chord of g between a_i and b_i, that is, $c_i = (g(b_i) - g(a_i))/(b_i - a_i)$, or $c_i = g'(a_i)$ if $a_i = b_i$. Then $c_i \geq 0$ because g is increasing. The slope of a chord of a convex function does not decrease when either endpoint moves to the right, and $a_i \geq a_{i+1}$, $b_i \geq b_{i+1}$; hence $c_1 \geq c_2 \geq \cdots \geq c_p$. Summation by parts gives

$$\sum_{i=1}^{p} (y_i^m - x_i^m) = \sum_{i=1}^{p} c_i(s_i - s_{i-1}) = \sum_{i=1}^{p-1} (c_i - c_{i+1})\, s_i + c_p s_p \geq 0.$$ □

Definition 10.18. For $X \in \mathbb{R}^{n \times n}$, let $|X| = (X^\top X)^{1/2}$, the SPSD square root of the SPSD matrix $X^\top X$: if $X^\top X = QDQ^\top$ with Q orthogonal and D diagonal, then $(X^\top X)^{1/2} = QD^{1/2}Q^\top$.

With the full singular value decomposition $A = \hat{U}\hat{\Lambda}\hat{V}^\top$, we get $A^\top A = \hat{V}\hat{\Lambda}^2\hat{V}^\top$, so $|A| = \hat{V}\hat{\Lambda}\hat{V}^\top$. In the same way, the polar decomposition $A = MP$ gives $AA^\top = MPP^\top M = M^2 = \hat{U}\hat{\Lambda}^2\hat{U}^\top$, and therefore $(AA^\top)^{1/2} = \hat{U}\hat{\Lambda}\hat{U}^\top$. Thus $\lambda(|A|) = \operatorname{sing}(A)$.

Corollary 10.19. *For every $X \in \mathbb{R}^{n\times n}$ and every integer $m \geq 1$,*

$$\mathrm{Tr}(X^m) \leq |\mathrm{Tr}(X^m)| \leq \mathrm{Tr}(|X|^m).$$

Proof. Let $\lambda_1, \dots, \lambda_n$ be the eigenvalues of X and $\sigma_1, \dots, \sigma_n$ its singular values. By the Schur decomposition, X^m is unitarily similar to an upper triangular matrix with diagonal $\lambda_1^m, \dots, \lambda_n^m$, and $|X|^m$ has eigenvalues $\sigma_1^m, \dots, \sigma_n^m$. The trace $\mathrm{Tr}(X^m)$ is real, and by Corollary 10.13 with $k = n$,

$$|\mathrm{Tr}(X^m)| = \Big|\sum_i \lambda_i^m\Big| \leq \sum_i |\lambda_i|^m \leq \sum_i \sigma_i^m = \mathrm{Tr}(|X|^m). \qquad \square$$

10.4 Lie product formula

Theorem 10.20 (Lie product formula)**.** *For all $A, B \in \mathbb{C}^{n\times n}$,*

$$e^{A+B} = \lim_{N\to\infty} \left(e^{A/N} e^{B/N}\right)^N. \tag{10.2}$$

To compare matrices we use norms.

Definition 10.21. A *matrix norm* on $\mathbb{C}^{n\times n}$ is a function $\|\cdot\| \colon \mathbb{C}^{n\times n} \to \mathbb{R}$ such that, for all A, B and all $c \in \mathbb{C}$,

1) $\|A\| \geq 0$, with equality only for $A = 0$,

2) $\|cA\| = |c|\|A\|$,

3) $\|A + B\| \leq \|A\| + \|B\|$,

4) $\|A \cdot B\| \leq \|A\|\|B\|$.

The spectral norm $\|A\| = \max_{x\neq 0} \|Ax\|_2/\|x\|_2 = \sigma_1(A)$ is a matrix norm with $\|I\| = 1$.

Claim 10.22. *Let $\|\cdot\|$ be a matrix norm, let $X, Y \in \mathbb{C}^{n\times n}$, and let $M = \max(\|X\|, \|Y\|)$. Then for every integer $N \geq 1$,*

$$\|X^N - Y^N\| \leq N M^{N-1} \|X - Y\|.$$

Proof. The difference telescopes:

$$\begin{aligned} X^N - Y^N &= (X^N - X^{N-1}Y) + (X^{N-1}Y - X^{N-2}Y^2) + \cdots + (XY^{N-1} - Y^N) \\ &= X^{N-1}(X - Y) + X^{N-2}(X - Y)Y + \cdots + (X - Y)Y^{N-1}. \end{aligned}$$

Each of the N terms has norm at most $M^{N-1}\|X - Y\|$, by the triangle inequality and submultiplicativity. $\square$

Lemma 10.23. *For a matrix norm with $\|I\| = 1$ and every A, $\|e^A\| \leq e^{\|A\|}$.*

Proof. By the triangle inequality, homogeneity and submultiplicativity,

$$\begin{aligned} \|e^A\| &= \Big\|I + A + \frac{A^2}{2!} + \cdots\Big\| \leq \|I\| + \|A\| + \frac{\|A^2\|}{2!} + \cdots \\ &\leq 1 + \|A\| + \frac{\|A\|^2}{2!} + \cdots = e^{\|A\|}. \end{aligned}$$ $\square$

Proof of the Lie product formula. Throughout, $\|\cdot\|$ is the spectral norm, and $O(N^{-2})$ denotes a matrix whose norm is at most CN^{-2}, for a constant C that depends only on A and B. We compare, as $N \to \infty$,

$$X_N = e^{(A+B)/N} \qquad \text{and} \qquad Y_N = e^{A/N} e^{B/N}.$$

Since $(A+B)/N$ commutes with itself, $X_N^N = e^{A+B}$, so we must show that $X_N^N - Y_N^N \to 0$. For every matrix Z, the tail of the exponential series satisfies $\|e^Z - I - Z\| \le \sum_{j\ge 2} \|Z\|^j/j! \le \|Z\|^2 e^{\|Z\|}$. By this Taylor expansion,

$$\begin{aligned} X_N &= I + \frac{A+B}{N} + O(N^{-2}), \\ Y_N &= \Big[I + \frac{A}{N} + O(N^{-2})\Big]\Big[I + \frac{B}{N} + O(N^{-2})\Big] = I + \frac{A}{N} + \frac{B}{N} + O(N^{-2}). \end{aligned}$$

Thus $X_N - Y_N = O(N^{-2})$. By Lemma 10.23,

$$\begin{aligned} \|X_N\| &\le e^{\|A+B\|/N} \le e^{(\|A\|+\|B\|)/N}, \\ \|Y_N\| &\le \|e^{A/N}\| \cdot \|e^{B/N}\| \le e^{(\|A\|+\|B\|)/N}. \end{aligned}$$

Therefore $M_N = \max(\|X_N\|, \|Y_N\|)$ satisfies $M_N^{N-1} \le e^{(N-1)(\|A\|+\|B\|)/N} \le e^{\|A\|+\|B\|}$. Finally, by Claim 10.22,

$$\|e^{A+B} - Y_N^N\| = \|X_N^N - Y_N^N\| \le N e^{\|A\|+\|B\|}\, O(N^{-2}) = O(1/N),$$

which tends to 0. □

10.5 Back to the proof of Golden–Thompson

Let A, B be symmetric, let $N \ge 1$, and let

$$X = e^{A/2^N}, \qquad Y = e^{B/2^N},$$

which are symmetric. We will show that

$$\operatorname{Tr}\big((XY)^{2^N}\big) \le \operatorname{Tr}\big(X^{2^N} \cdot Y^{2^N}\big) : \tag{10.3}$$

the X's and Y's can be unmixed. This proves Golden–Thompson. Indeed, $X^{2^N} = e^A$ and $Y^{2^N} = e^B$, so the right side of (10.3) is $\operatorname{Tr}(e^A e^B)$. By the Lie product formula (10.2), applied along $N' = 2^N$, the matrices $(XY)^{2^N}$ converge to e^{A+B}, and the trace is continuous, so $\operatorname{Tr}(e^{A+B}) = \lim_{N\to\infty} \operatorname{Tr}\big((XY)^{2^N}\big)$.

Lemma 10.24. *For all symmetric $X, Y \in \mathbb{R}^{n\times n}$ and every integer $j \ge 1$,*

$$\operatorname{Tr}\big((XY)^{2^j}\big) \le \operatorname{Tr}\big((X^2Y^2)^{2^{j-1}}\big).$$

Proof. By Corollary 10.19, the corollary to Weyl's theorem, applied to the real matrix XY,

$$\operatorname{Tr}\big((XY)^{2^j}\big) \le \big|\operatorname{Tr}\big[(XY)^{2^j}\big]\big| \le \operatorname{Tr}\big[|XY|^{2^j}\big] = \operatorname{Tr}\big[(YXXY)^{2^{j-1}}\big],$$

where the last step uses $|XY|^2 = (XY)^\top XY = YXXY$, since X and Y are symmetric. Writing out the product and moving the first factor Y to the end, which does not change the trace since $\operatorname{Tr}(UV) = \operatorname{Tr}(VU)$,

$$\operatorname{Tr}\big[YX^2Y\,YX^2Y\cdots YX^2Y\big] = \operatorname{Tr}\big[X^2Y^2\,X^2Y^2\cdots X^2Y^2\big] = \operatorname{Tr}\big((X^2Y^2)^{2^{j-1}}\big).$$ □

Proof of Theorem 10.1. Apply Lemma 10.24 repeatedly, to the symmetric pairs (X^{2^i}, Y^{2^i}) with $j = N - i$, for $i = 0, 1, \ldots, N-1$:

$$\mathrm{Tr}\big((XY)^{2^N}\big) \le \mathrm{Tr}\big((X^2Y^2)^{2^{N-1}}\big) \le \mathrm{Tr}\big((X^4Y^4)^{2^{N-2}}\big) \le \cdots \le \mathrm{Tr}\big(X^{2^N}Y^{2^N}\big).$$

This is (10.3), which proves the theorem as explained above. □

11 Matrix Chernoff Bounds (Ahlswede–Winter, Tropp)

Lecture of February 10, 2020. No Spring 2020 notes are available for this lecture, so it is based on Gary L. Miller's Spring 2021 handwritten notes. The lecture as given in Spring 2020 may have differed.

A Chernoff bound says that a sum of independent bounded random variables is exponentially unlikely to be much larger than its mean. This lecture first proves such a bound for sums of independent 0-1 random variables, using the Laplace transform and Markov's inequality. It then states Tropp's matrix Chernoff bound, which gives the same kind of control over the extreme eigenvalues of a sum of independent random positive semidefinite matrices, and shows that it implies the Ahlswede–Winter theorem of Lecture 8. Two examples prepare the use of these bounds for sampling: one with diagonal matrices and one with the edge Laplacians of a graph.

11.1 Chernoff bounds and matrix Chernoff bounds

We start with a simple Chernoff bound.

Theorem 11.1 (Simple Chernoff bound)**.** *Let $X_1, \dots, X_n$ be independent and identically distributed* 0-1 *random variables with $\mathbb{E}[X_i] = p$, where $0 < p \le 1$, and let $X = \sum_i X_i$, so that $\mathbb{E}[X] = \mu = pn$. Then for every $\beta \ge 1$,*

$$\Pr[X \ge (1+\beta)\mu] < e^{-\beta\mu/3}.$$

The proof combines the Laplace transform of X, that is, its moment generating function $t \mapsto \mathbb{E}[e^{tX}]$, with Markov's inequality.

Fact 11.2 (Markov's inequality)**.** *If X is a nonnegative random variable and $a > 0$, then*

$$\Pr[X \ge a] \le \frac{\mathbb{E}[X]}{a}.$$

Proof of Theorem 11.1. Let $t > 0$ and $m > 0$. Then

$$\begin{aligned}
\Pr[X \ge m] &= \Pr\left[e^{tX} \ge e^{tm}\right] && (e^x \text{ is strictly increasing}) \\
&\le e^{-tm}\mathbb{E}\left[e^{tX}\right] && (\text{Markov}) \\
&= e^{-tm}\mathbb{E}\left[\prod_i e^{tX_i}\right] && (X = X_1 + \cdots + X_n) \\
&= e^{-tm}\prod_i \mathbb{E}\left[e^{tX_i}\right] && (\text{independence of the } X_i) \\
&= e^{-tm}\left(e^t p + 1 - p\right)^n && (\Pr[X_i = 1] = p).
\end{aligned}$$

Thus

$$\Pr[X \ge m] \le e^{-tm}\left(e^t p + 1 - p\right)^n \qquad \text{for all } t, m > 0. \tag{11.1}$$

Now set $m = (1+\beta)pn$ and choose t to make the right side small. Since $1 + y \le e^y$ for all real y, with strict inequality for $y \ne 0$, we have $e^t p + 1 - p < e^{p(e^t - 1)}$ for $t > 0$. With $t = \ln(1+\beta)$, the right side of (11.1) is therefore less than

$$e^{-(1+\beta)\ln(1+\beta)\,pn}\, e^{\beta pn} = \left(\frac{e^\beta}{(1+\beta)^{1+\beta}}\right)^{pn}.$$

Thus

$$\Pr[X \geq (1+\beta)\mu] < \left(\frac{e^{\beta}}{(1+\beta)^{1+\beta}}\right)^{\mu}. \tag{11.2}$$

To get a weaker but simpler bound, we bound $\ln(1+x)$ from below. The Taylor series of $f(x) = \ln(1+x)$ at 0 comes from

$$\begin{aligned} f(x) &\qquad & f(0) &= 0 \\ f'(x) &= \frac{1}{1+x} & f'(0) &= 1 \\ f''(x) &= -\frac{1}{(1+x)^2} & f''(0) &= -1 \\ f'''(x) &= \frac{2}{(1+x)^3} & f'''(0) &= 2, \end{aligned}$$

so that

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \cdots$$

for $|x| < 1$. Truncating after the second term gives $\ln(1+x) > x - x^2/2$ for $x > 0$, that is, $1 + x > e^{x - x^2/2}$, and hence

$$(1+\beta)^{1+\beta} > e^{(\beta-\beta^2/2)(1+\beta)} = e^{\beta+\beta^2/2-\beta^3/2}.$$

The cubic term makes this bound useless for $\beta \geq 1$. We use instead the rational lower bound

$$\ln(1+x) \geq \frac{2x}{2+x} \qquad \text{for all } x \geq 0. \tag{11.3}$$

Since $2x/(2+x) = x - x^2/2 + x^3/4 - \cdots$ for $|x| < 2$, the right side agrees with the Taylor series up to the x^2 term. To prove it, let $h(x) = \ln(1+x) - 2x/(2+x)$. Then $h(0) = 0$ and

$$h'(x) = \frac{1}{1+x} - \frac{4}{(2+x)^2} = \frac{x^2}{(1+x)(2+x)^2} \geq 0.$$

By (11.3),

$$\beta - (1+\beta)\ln(1+\beta) \leq \beta - \frac{2\beta(1+\beta)}{2+\beta} = -\frac{\beta^2}{2+\beta},$$

so $(1+\beta)^{1+\beta} \geq e^{\beta+\beta^2/(2+\beta)}$. Hence, by (11.2),

$$\Pr[X \geq (1+\beta)\mu] < e^{-\beta^2\mu/(2+\beta)}.$$

For $\beta \geq 1$ we have $\beta/(2+\beta) \geq 1/3$, so the right side is at most $e^{-\beta\mu/3}$, which proves the theorem. For $0 < \beta \leq 1$ we have $2 + \beta \leq 3$, and the same bound gives $\Pr[X \geq (1+\beta)\mu] < e^{-\beta^2\mu/3}$. □

Remark 11.3. The step $\mathbb{E}[e^{t(X_1+\cdots+X_n)}] = \mathbb{E}[\prod_i e^{tX_i}]$ uses $e^{a+b} = e^a e^b$. It is not true if the X_i are matrices, since $e^{A+B} \neq e^A e^B$ in general (Lecture 10).

For random matrices, the analogue of the Chernoff bound controls the extreme eigenvalues of a sum. Here $\preceq$ is the Loewner order (Lecture 9), and $\lambda_{\min}$, $\lambda_{\max}$ denote the smallest and largest eigenvalues.

Theorem 11.4 (Tropp [Tro12]). *Let $X_1, \dots, X_k$ be independent random symmetric $d \times d$ matrices with $0 \preceq X_i \preceq RI$ for some $R > 0$. Suppose that*

$$\mu_{\min} \cdot I \preceq \sum_i \mathbb{E}[X_i] \preceq \mu_{\max} \cdot I.$$

Then:

(+) *for every $\delta \geq 0$,*

$$\Pr\Big[\lambda_{\max}\Big(\sum_i X_i\Big) \geq (1+\delta)\mu_{\max}\Big] \leq d \cdot \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu_{\max}/R} \leq d \begin{cases} e^{-\delta^2 \mu_{\max}/3R} & \text{for } \delta \leq 1, \\ e^{-\delta \mu_{\max}/3R} & \text{for } \delta > 1; \end{cases}$$

(−) *for every δ with $0 \leq \delta < 1$,*

$$\Pr\Big[\lambda_{\min}\Big(\sum_i X_i\Big) \leq (1-\delta)\mu_{\min}\Big] \leq d \cdot \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu_{\min}/R} \leq d\, e^{-\delta^2 \mu_{\min}/2R}.$$

We state Tropp's theorem without proof; see [Tro12]. The second inequality in each line is elementary. For (+), it is the computation at the end of the proof of Theorem 11.1, with δ for β.

11.2 Tropp implies Ahlswede–Winter

We show that Tropp's theorem implies the Ahlswede–Winter theorem (AW) of Lecture 8. The AW hypotheses on the random $n \times n$ SPSD matrices $Y_1, \dots, Y_k$ are:

1) the Y_i are independent and identically distributed;

2) $\mathbb{E}[Y_i] = Z$;

3) $Y_i \preceq \mu Z$.

Hypothesis 1) gives the independence that Tropp's theorem needs. To match hypotheses 2) and 3) with Tropp's, we normalize. Assume that Z is positive definite, so that $Z^{-1/2}$ exists, and set $\bar{Y}_i = Z^{-1/2} Y_i Z^{-1/2}$. For a random matrix Y_i with finitely many values, linearity of expectation gives

$$\begin{aligned} \mathbb{E}[\bar{Y}_i] &= \sum_Y \Pr[Y_i = Y]\, Z^{-1/2} Y Z^{-1/2} \\ &= Z^{-1/2}\Big(\sum_Y \Pr[Y_i = Y]\, Y\Big) Z^{-1/2} \\ &= Z^{-1/2} Z Z^{-1/2} = I. \end{aligned}$$

For 3), note that the congruence $X \mapsto Z^{-1/2} X Z^{-1/2}$ preserves the Loewner order: for every x, the vector $y = Z^{-1/2}x$ satisfies $x^\top Z^{-1/2} X Z^{-1/2} x = y^\top X y$. Hence

$$\bar{Y}_i = Z^{-1/2} Y_i Z^{-1/2} \preceq Z^{-1/2} \mu Z Z^{-1/2} = \mu I.$$

Also $\bar{Y}_i \succeq 0$, by the same congruence.

Now apply Tropp's theorem to $\bar{Y}_1, \dots, \bar{Y}_k$, with $d = n$, $R = \mu$ and $\mu_{\min} = \mu_{\max} = k$, since $\sum_i \mathbb{E}[\bar{Y}_i] = kI$. For $0 < \varepsilon \le 1$, the event $(1-\varepsilon)kI \preceq \sum_i \bar{Y}_i \preceq (1+\varepsilon)kI$ fails only if $\lambda_{\max}(\sum_i \bar{Y}_i) > (1+\varepsilon)k$ or $\lambda_{\min}(\sum_i \bar{Y}_i) < (1-\varepsilon)k$. For $\varepsilon = 1$ the second case is impossible, because $\sum_i \bar{Y}_i \succeq 0$. By $(+)$ and $(-)$ and a union bound, this has probability at most $ne^{-\varepsilon^2 k/3\mu} + ne^{-\varepsilon^2 k/2\mu} \le 2ne^{-\varepsilon^2 k/3\mu}$. Conjugating by $Z^{1/2}$ turns the event into $(1-\varepsilon)Z \preceq \sum_i Y_i/k \preceq (1+\varepsilon)Z$. This is the AW theorem, with the constant $c = 1/3$.

Remark 11.5. If Z is singular, as in Example 11.7 below, where $Z = L$, the same argument works on the range of Z, with the pseudo-inverse $(Z^\dagger)^{1/2}$ (Lecture 9) in place of $Z^{-1/2}$: the hypotheses $0 \preceq Y_i \preceq \mu Z$ force every Y_i to vanish on the kernel of Z. On the range of Z, a space of dimension $\operatorname{rank} Z$ ($= n-1$ for the Laplacian of a connected graph), we then have $\mathbb{E}[\bar{Y}_i] = I$ and $0 \preceq \bar{Y}_i \preceq \mu I$, so Tropp's theorem applies there with $d = \operatorname{rank} Z$. On all of $\mathbb{R}^n$, by contrast, $\mathbb{E}[\bar{Y}_i]$ is the orthogonal projection onto the range of Z, and $\lambda_{\min}(\sum_i \bar{Y}_i) = 0$ always.

11.3 Examples

Example 11.6. Let E_i be the $d \times d$ matrix with a single 1 in diagonal position i and zeros elsewhere:

$$E_i = \begin{pmatrix} 0 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 0 \end{pmatrix}.$$

Let the random matrix Y be obtained by picking $i \in \{1, \dots, d\}$ uniformly at random and returning dE_i. Then

$$\mathbb{E}[Y] = \sum_{i=1}^{d} \frac{1}{d}\, dE_i = \sum_i E_i = I.$$

Let $Y_1, \dots, Y_k$ be independent copies of Y. Then

$$\sum_{i=1}^{k} \mathbb{E}[Y_i] = kI, \qquad \text{so} \qquad kI \preceq \sum_{i=1}^{k} \mathbb{E}[Y_i] \preceq kI.$$

Each Y_i satisfies $0 \preceq Y_i \preceq dI$. By Tropp's bound $(+)$ with $R = d$, $\mu_{\max} = k$ and $\delta = 2$ (the case $\delta > 1$),

$$\Pr\Big[\lambda_{\max}\Big(\sum_i Y_i\Big) \ge (1+2)k\Big] \le d\, e^{-2k/3d}. \tag{$*$}$$

Set $k = c\, d \ln d$ (rounded up to an integer, which only decreases the bound). Then the right side of $(*)$ is

$$d\, e^{-2cd \ln d/3d} = d \cdot d^{-2c/3} = d^{1-2c/3} = d^{-1} \qquad \text{for } c = 3.$$

Thus, with $k = 3d \ln d$ samples, $\lambda_{\max}(\sum_i Y_i) < 3k$ with probability at least $1 - 1/d$.

Example 11.7. Let $G = (V, E)$ be a graph with unit weights, m edges and Laplacian $L = L(G)$. For an edge $e = (i, j)$, let

$$H_e = H_{ij} = \begin{pmatrix} \ddots & & & \\ & 1 & -1 & \\ & -1 & 1 & \\ & & & \ddots \end{pmatrix}$$

be its edge Laplacian, with the nonzero entries in rows and columns i and j. Then

$$L = \sum_{e \in E} H_e.$$

Let the random matrix Y be obtained by picking an edge $e \in E$ uniformly at random and returning mH_e. Then

$$\mathbb{E}[Y] = \sum_{e \in E} \frac{1}{m} mH_e = \sum_{e \in E} H_e = L.$$

12 Graph Cuts and Eigenvalues: Cheeger Inequality

Lecture of February 12, 2020. Based on Gary L. Miller's handwritten notes.

A cut of a graph is good when little edge weight separates two large parts of the vertex set. This lecture defines several measures of cut quality (the quotient cut, the sparsest cut and the conductance) and relates them to the second-smallest eigenvalue of a Laplacian. The main results are Cheeger's inequalities, which bound the quotient cut and the conductance above and below in terms of λ_2. We prove the easy direction of the first using the Courant–Fischer theorem (Lecture 9), and a strengthened form of the hard direction of the second, for Laplacians with vertex masses. Its proof is constructive: from any test vector it produces a good threshold cut. The bug graph shows that the most obvious threshold cut, the one that splits the vertices in half, can be far from the best.

Throughout this lecture, $G = (V, E, w)$ is a connected undirected graph with positive edge weights $w(a, b) = w_{ab}$ and $n \geq 2$ vertices, $V = \{v_1, \ldots, v_n\}$. The weighted degree of v_i is d_i, $D = \operatorname{diag}(d_1, \ldots, d_n)$, A is the adjacency matrix, $L = D - A$ is the Laplacian, and $\Delta = \max_i d_i$. For $S \subseteq V$ we write $\bar{S} = V \setminus S$. Since G is connected and $n \geq 2$, every d_i is positive.

12.1 Graph cuts and eigenvalues

There are two types of graph cuts: *edge cuts*, which remove edges to disconnect the graph, and *vertex cuts*, which remove vertices to disconnect the graph. We consider only edge cuts. Given G, we look for a partition of V into a set S and its complement $\bar{S}$ such that little edge weight crosses between them, relative to the sizes of the two sides.

Definition 12.1. For $A, B \subseteq V$, let

$$\operatorname{Cap}(A, B) \coloneqq \sum_{a \in A} \sum_{b \in B} w(a, b),$$

where $w(a, b) = 0$ if $(a, b) \notin E$. For disjoint A and B, this is the total weight of the edges between A and B. We also write

$$\operatorname{Cap}(A) \coloneqq \operatorname{Cap}(A, V) = \sum_{a \in A} d_a.$$

12.2 Cut measures

Each of the following measures is a minimum over all cuts, that is, over all S with $\emptyset \neq S \subsetneq V$; the excluded choices $S = \emptyset$ and $S = V$ would give $0/0$.

Definition 12.2.

1) The *quotient cut*, or *isoperimetric number*, of G is

$$q(G) = \min_{\emptyset \neq S \subsetneq V} \frac{\operatorname{Cap}(S, \bar{S})}{\min\{|S|, |\bar{S}|\}}.$$

2) The *sparsest cut* of G is

$$\alpha(G) = \min_{\emptyset \neq S \subsetneq V} \frac{\operatorname{Cap}(S, \bar{S})}{|S| \cdot |\bar{S}|}.$$

Proposition 12.3. $\frac{n}{2}\,\alpha(G) \le q(G) \le n\,\alpha(G)$.

Proof. For every cut S, we have $|S|\cdot|\bar{S}| = \min\{|S|,|\bar{S}|\}\cdot\max\{|S|,|\bar{S}|\}$ and $n/2 \le \max\{|S|,|\bar{S}|\} \le n$. Hence the quotient of S in the definition of q lies between $n/2$ and n times its quotient in the definition of α. Taking minima over S preserves both inequalities. □

Definition 12.4.

4) The *conductance* of G is

$$\Phi(G) = \min_{\emptyset \neq S \subsetneq V} \frac{\mathrm{Cap}(S,\bar{S})}{\min\{\mathrm{Cap}(S),\mathrm{Cap}(\bar{S})\}}.$$

5) The *sparsest cut* with degree demands is

$$\bar{\alpha}(G) = \min_{\emptyset \neq S \subsetneq V} \frac{\mathrm{Cap}(S,\bar{S})}{\mathrm{Cap}(S)\cdot\mathrm{Cap}(\bar{S})}.$$

We can view the sparsest cut as a minimum cut relative to demands. Suppose that the traffic, or demand, between v_i and v_j is $d_i \cdot d_j$, and let

$$\mathrm{Traffic}(S,\bar{S}) \coloneqq \sum_{v_i\in S,\ v_j\notin S} d_i d_j$$

be the demand that must cross the cut. This factors:

$$\mathrm{Traffic}(S,\bar{S}) = \Big(\sum_{v_i\in S} d_i\Big)\Big(\sum_{v_j\in\bar{S}} d_j\Big) = \mathrm{Cap}(S)\,\mathrm{Cap}(\bar{S}).$$

We set $\mathrm{Demand}(S,\bar{S}) \coloneqq \mathrm{Traffic}(S,\bar{S})$, so that $\bar{\alpha}(G)$ is the minimum over cuts of capacity divided by demand.

Definition 12.5. Let $f\colon V \to \mathbb{R}_{>0}$. The *product demand* defined by f is $\mathrm{Demand}_f(v_i,v_j) = f(v_i)\cdot f(v_j)$, and for a cut S,

$$\mathrm{Demand}_f(S,\bar{S}) = \sum_{v_i\in S,\ v_j\notin S} f(v_i)f(v_j) = f(S)f(\bar{S}), \qquad \text{where } f(S) = \sum_{v\in S} f(v).$$

The sparsest cut for this demand is

$$\bar{\alpha}_f(G) \coloneqq \min_{\emptyset \neq S \subsetneq V} \frac{\mathrm{Cap}(S,\bar{S})}{\mathrm{Demand}_f(S,\bar{S})}.$$

For $f = d$ this is $\bar{\alpha}(G)$, and for $f \equiv 1$ it is $\alpha(G)$. The conductance corresponds to the vertex weights d_i, which we can also view as masses.

Definition 12.6. Let $m_1,\dots,m_n > 0$ be *masses* on the vertices, and let $\mathrm{Mass}(S) = \sum_{v_i\in S} m_i$. The *conductance of G with respect to the masses* is

$$\Phi_m(G) = \min_{\emptyset \neq S \subsetneq V} \frac{\mathrm{Cap}(S,\bar{S})}{\min\{\mathrm{Mass}(S),\mathrm{Mass}(\bar{S})\}}.$$

With $m_i = d_i$ we have $\mathrm{Mass}(S) = \mathrm{Cap}(S)$, so $\Phi_m(G) = \Phi(G)$. Scaling all masses by a constant $c > 0$ divides Φ_m by c, so the normalized masses $p_i = d_i/\sum_j d_j$ give $\Phi_p(G) = (\sum_j d_j)\,\Phi(G)$. Since $\sum_i p_i = 1$, the demand and mass versions differ by at most a factor of 2, as α and q do in Proposition 12.3: $p(S)p(\bar{S}) = \min\cdot\max$ with $1/2 \le \max\{p(S),p(\bar{S})\} \le 1$, so $\Phi_p(G) \le \bar{\alpha}_p(G) \le 2\,\Phi_p(G)$.

12.3 Main Cheeger theorems

Theorem 12.7 (Cheeger's inequality for the quotient cut)**.** *Let $\lambda_2 = \lambda_2(L)$. Then*

$$\frac{\lambda_2}{2} \le q(G) \le \sqrt{2\Delta\lambda_2}.$$

Theorem 12.8 (Cheeger's inequality for the conductance)**.** *Let $\mathcal{L} = D^{-1/2}LD^{-1/2}$ be the normalized Laplacian of G, and let $\lambda_2 = \lambda_2(\mathcal{L})$ be its second-smallest eigenvalue. Then*

$$\frac{\lambda_2}{2} \le \Phi(G) \le \sqrt{2\lambda_2}.$$

The upper bounds on $q(G)$ and $\Phi(G)$ are called the *Cheeger direction*. Both follow from a version for Laplacians with vertex masses, as in the spring-mass Laplacian $Lx = \lambda Mx$.

Theorem 12.9. *Let $m_1, \dots, m_n > 0$ be masses and $M = \operatorname{diag}(m_1, \dots, m_n)$. Suppose that $x \neq 0$ satisfies $x^\top M\mathbf{1} = 0$ and has Rayleigh quotient $\frac{x^\top Lx}{x^\top Mx} = \lambda$. Then*

$$\Phi_m(G) \le \sqrt{2 \max_i \Big\{ \frac{d_i}{m_i} \Big\} \lambda}.$$

Theorem 12.9 is the case of Theorem 12.14 below that only asserts the existence of a good cut.

The normalized Laplacian is related to the masses $m = d$ as follows.

Lemma 12.10. $\lambda_2(\mathcal{L}) = \min\{x^\top Lx / x^\top Dx \mid x \neq 0,\ x^\top D\mathbf{1} = 0\}$.

Proof. The matrix $\mathcal{L}$ is symmetric, and it is positive semidefinite because every u satisfies

$$u^\top \mathcal{L}u = (D^{-1/2}u)^\top L(D^{-1/2}u) \ge 0.$$

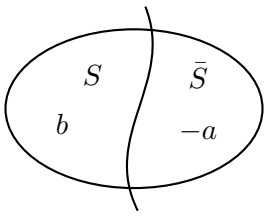
Moreover $\mathcal{L}D^{1/2}\mathbf{1} = D^{-1/2}L\mathbf{1} = 0$. So 0 is its smallest eigenvalue, with eigenvector $D^{1/2}\mathbf{1}$. As in Lecture 9, $\lambda_2(\mathcal{L})$ is the minimum of $u^\top \mathcal{L}u / u^\top u$ over nonzero $u \perp D^{1/2}\mathbf{1}$. Substituting $u = D^{1/2}x$ gives $u^\top \mathcal{L}u = x^\top Lx$, $u^\top u = x^\top Dx$ and $u^\top D^{1/2}\mathbf{1} = x^\top D\mathbf{1}$. □

12.4 The easy direction for the quotient cut

Claim 12.11. *$\lambda_2(L) \le 2 \cdot q(G)$; that is, there is a nonzero vector x such that $x^\top\mathbf{1} = 0$ and $\dfrac{x^\top Lx}{x^\top x} \le 2 \cdot q(G)$.*

Proof. Let S be a cut with $|S| \le n/2$, and set $a = |S|$ and $b = |\bar{S}|$, so that $a + b = n$ and $b \ge n/2$. Let

$$x_i = \begin{cases} b & \text{if } v_i \in S, \\ -a & \text{otherwise.} \end{cases}$$



Then

$$x^\top \mathbf{1} = b \cdot |S| - a|\bar{S}| = ab - ab = 0.$$

Only the edges between S and $\bar{S}$ contribute to $x^\top L x = \sum_{(i,j)\in E} w_{ij}(x_i - x_j)^2$, each with $(x_i - x_j)^2 = (a+b)^2$. Hence

$$\frac{x^\top L x}{x^\top x} = \frac{(a+b)^2 \operatorname{Cap}(S,\bar{S})}{ab^2 + ba^2} = \frac{n^2 \operatorname{Cap}(S,\bar{S})}{n\,ab} = \frac{\operatorname{Cap}(S,\bar{S})}{a}\Big(\frac{n}{b}\Big) \le 2\,\frac{\operatorname{Cap}(S,\bar{S})}{|S|},$$

since $n/b \le 2$. Now choose S to attain the minimum in $q(G)$; exchanging S and $\bar{S}$ if necessary, which does not change the quotient, we may assume $|S| \le n/2$. Then $\operatorname{Cap}(S,\bar{S})/|S| = q(G)$, and by Courant–Fischer (Lecture 9), $\lambda_2(L) \le x^\top L x / x^\top x \le 2\,q(G)$. □

Remark 12.12. The same computation proves the lower bound of Theorem 12.8. Let S attain the minimum in $\Phi(G)$, and let $x_i = \operatorname{Cap}(\bar{S})$ for $v_i \in S$ and $x_i = -\operatorname{Cap}(S)$ otherwise. Then $x^\top D\mathbf{1} = 0$, $x^\top L x = \operatorname{Cap}(V)^2 \operatorname{Cap}(S,\bar{S})$ and $x^\top D x = \operatorname{Cap}(S)\operatorname{Cap}(\bar{S})\operatorname{Cap}(V)$. Since $\operatorname{Cap}(V) \le 2\max\{\operatorname{Cap}(S), \operatorname{Cap}(\bar{S})\}$, Lemma 12.10 gives

$$\lambda_2(\mathcal{L}) \le \frac{\operatorname{Cap}(V)\ \operatorname{Cap}(S,\bar{S})}{\operatorname{Cap}(S)\operatorname{Cap}(\bar{S})} \le \frac{2\ \operatorname{Cap}(S,\bar{S})}{\min\{\operatorname{Cap}(S), \operatorname{Cap}(\bar{S})\}} = 2\,\Phi(G).$$

12.5 The upper bound for the conductance

We prove a stronger version, for a spring-mass Laplacian $Lx = \lambda M x$ with masses $m_1, \dots, m_n > 0$ and $M = \operatorname{diag}(m_1, \dots, m_n)$. Let

$$\delta = \max_i \Big(\frac{d_i}{m_i}\Big).$$

The proof is effective: given a vector x, it finds a good cut among the threshold cuts of x.

Definition 12.13. Let $f\colon V \to \mathbb{R}$, and number the vertices so that $f(v_i) \ge f(v_{i+1})$ for all i, breaking ties arbitrarily. For $0 \le i \le n$, let $S_i = \{v_1, \dots, v_i\}$, so that $S_0 = \emptyset$ and $S_n = V$. For $1 \le i \le n-1$, the cut S_i is the *ith threshold cut* of f.

Theorem 12.14. *Let $x \in \mathbb{R}^n$ be nonzero with $x^\top \begin{pmatrix} m_1 \\ \vdots \\ m_n \end{pmatrix} = 0$. Then some threshold cut S_i of x, with $1 \le i \le n-1$, satisfies*

$$\frac{\operatorname{Cap}(S_i,\bar{S}_i)}{\min\{\operatorname{Mass}(S_i), \operatorname{Mass}(\bar{S}_i)\}} \le \sqrt{2\delta\,\frac{x^\top L x}{x^\top M x}}.$$

In particular,

$$\Phi_m(G) = \min_{\emptyset \ne S \subsetneq V} \frac{\operatorname{Cap}(S,\bar{S})}{\min\{\operatorname{Mass}(S), \operatorname{Mass}(\bar{S})\}} \le \sqrt{2\delta\,\frac{x^\top L x}{x^\top M x}}.$$

The proof occupies the rest of the lecture. Algorithmically, it suffices to sort the vertices by x and evaluate the $n-1$ threshold cuts.

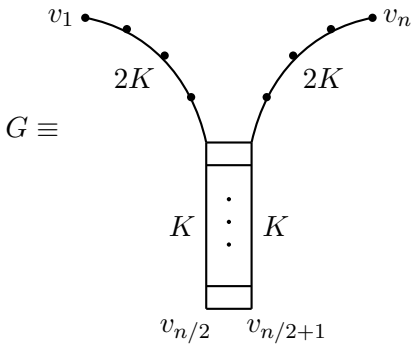
Corollary 12.15. *The upper bounds in Theorems 12.7 and 12.8 hold.*

Proof. For Theorem 12.7, take $m_i = 1$ for all i, so that $M = I$, $\mathrm{Mass}(S) = |S|$, $\Phi_m(G) = q(G)$ and $\delta = \Delta$. Let x be an eigenvector of L for $\lambda_2(L)$ orthogonal to $\mathbf{1}$ (Lecture 9). Then $x^\top M\mathbf{1} = 0$ and $x^\top Lx / x^\top x = \lambda_2(L)$, so $q(G) \le \sqrt{2\Delta\lambda_2(L)}$ by Theorem 12.14.

For Theorem 12.8, take $m_i = d_i$, so that $M = D$, $\Phi_m(G) = \Phi(G)$ and $\delta = 1$. By Lemma 12.10, some nonzero x with $x^\top D\mathbf{1} = 0$ has $x^\top Lx / x^\top Dx = \lambda_2(\mathcal{L})$, so $\Phi(G) \le \sqrt{2\lambda_2(\mathcal{L})}$. □

Not every threshold cut is good. In particular, the threshold cut $S_{n/2}$, which splits the vertices in half, may be bad, even for an eigenvector of λ_2.

Example 12.16 (The bug graph)**.** Let $K \ge 2$. The bug graph G consists of a ladder with K rungs, whose two sides are paths of K vertices, and two *ears*, each a path of $2K$ vertices attached to one of the top corners of the ladder. All weights are 1, and $n = 6K$:



The left half of G (the left ear and the left side of the ladder) is a path of $3K$ vertices from v_1 to $v_{n/2}$, and the right half is its mirror image. The graph has an automorphism σ, the reflection that exchanges the two halves.

Claim 12.17. *Let $f \neq 0$ satisfy $Lf = \lambda_2 f$ for the bug graph. Then f is odd with respect to σ, that is, $f(v) = -f(\sigma(v))$ for every vertex v.*

The proof of Claim 12.17 is left as homework.

Claim 12.18. *For the bug graph, λ_2 is a simple eigenvalue, and an eigenvector f for it has no zero entry, with one sign on the left half and the opposite sign on the right half.*

Proof. Write $u_i = v_i$ for $1 \le i \le 3K$, so that $u_1, \dots, u_{3K}$ is the left half in path order, and order the vertices as $u_1, \dots, u_{3K}, \sigma(u_1), \dots, \sigma(u_{3K})$. Let L_P be the Laplacian of the path $u_1 u_2 \cdots u_{3K}$, and let R be the diagonal $3K \times 3K$ matrix with $R_{ii} = 1$ if u_i lies on the ladder and $R_{ii} = 0$ otherwise. The rungs join u_i to $\sigma(u_i)$ for the ladder vertices u_i, so in this order

$$L = \begin{pmatrix} L_P + R & -R \\ -R & L_P + R \end{pmatrix}, \qquad \text{and hence} \qquad L \begin{pmatrix} g \\ -g \end{pmatrix} = \begin{pmatrix} Mg \\ -Mg \end{pmatrix} \quad \text{for every } g \in \mathbb{R}^{3K},$$

where $M = L_P + 2R$. Let μ_1 be the smallest eigenvalue of M, and let c be the largest diagonal entry of M. The matrix $cI - M$ has nonnegative entries, and its graph $G(cI - M)$ contains both directions of every edge of the path $u_1 \cdots u_{3K}$, so it is strongly connected. By the Perron–Frobenius theorem (Theorem 5.13), its spectral radius is a simple eigenvalue, and every eigenvector for it has no zero entry and all entries of one sign. Since $cI - M$ is symmetric, its spectral radius

is its largest eigenvalue, $c - \mu_1$. Hence μ_1 is a simple eigenvalue of M, and an eigenvector g_1 for it has no zero entry and all entries of one sign.

Now $(g_1; -g_1)$ is an eigenvector of L with eigenvalue μ_1, and it is orthogonal to $\mathbf{1}$, so $\lambda_2 \le \mu_1$ by the Courant–Fischer theorem (Lecture 9). Conversely, let $f \neq 0$ satisfy $Lf = \lambda_2 f$. By Claim 12.17, $f = (g; -g)$ for some $g \neq 0$, and then $Mg = \lambda_2 g$. Thus λ_2 is an eigenvalue of M, so $\lambda_2 \ge \mu_1$. Hence $\lambda_2 = \mu_1$, and g is a multiple of g_1. So every eigenvector for λ_2 is a multiple of $(g_1; -g_1)$, which proves the claim. □

By Claim 12.18, we may take an eigenvector f for λ_2 with $f > 0$ on the left half and $f < 0$ on the right half. Sorting by f, the threshold cut $S_{n/2}$ is the left half. It cuts the K rungs, so $\mathrm{Cap}(S_{n/2}, \bar{S}_{n/2}) = K$, and its quotient is $\frac{K}{3K} = \frac{1}{3}$. A better cut removes one ear: for S the left ear, $\mathrm{Cap}(S, \bar{S}) = 1$ and $|S| = 2K$, so its quotient is $\frac{1}{2K} = \frac{3}{n}$, and, since $\mathrm{Cap}(S) = 4K - 1$ is less than $\mathrm{Cap}(\bar{S}) = 10K - 3$, its conductance is $1/\mathrm{Cap}(S) = \frac{1}{4K-1}$. Thus $q(G) \le 3/n$, which is less than the quotient $1/3$ of the threshold cut $S_{n/2}$, and far less when K is large.

12.6 The proof: preliminaries

The Laplacian $L = D - A$ satisfies

$$z^\top L z = \sum_{(i,j)\in E} w_{ij}(z_i - z_j)^2,$$

with each edge counted once. The *sum-Laplacian* is $P = D + A$.

Claim 12.19. *$z^\top P z = \sum_{(i,j)\in E} w_{ij}(z_i + z_j)^2$, and $P = 2D - L$.*

Proof. We have $z^\top D z = \sum_i d_i z_i^2 = \sum_{(i,j)\in E} w_{ij}(z_i^2 + z_j^2)$ and $z^\top A z = \sum_{(i,j)\in E} 2 w_{ij} z_i z_j$; adding gives the first identity. The second follows from $A = D - L$. □

Lemma 12.20. *Let L and P be the Laplacian and the sum-Laplacian of any graph on V with positive weights w_{ij}. For every $z \in \mathbb{R}^n$,*

$$(z^\top L z)(z^\top P z) \ge \Big(\sum_{(i,j)\in E} w_{ij} |z_i^2 - z_j^2| \Big)^2.$$

Proof. Let a and b be the vectors indexed by the edges with entries

$$a_{ij} = \sqrt{w_{ij}}\,|z_i - z_j|, \qquad b_{ij} = \sqrt{w_{ij}}\,|z_i + z_j|.$$

By the Cauchy–Schwarz inequality $(a^\top b)^2 \le (a^\top a)(b^\top b)$,

$$\begin{aligned}(z^\top L z)(z^\top P z) &= \Big(\sum w_{ij}|z_i - z_j|^2\Big)\Big(\sum w_{ij}|z_i + z_j|^2\Big) \\ &\ge \Big(\sum w_{ij}|z_i - z_j|\,|z_i + z_j|\Big)^2 = \Big(\sum w_{ij}|z_i^2 - z_j^2|\Big)^2.\end{aligned}$$

□

12.7 The proof

Proof of Theorem 12.14. Write $z = x$, so that $z \neq 0$ and $z^\top M\mathbf{1} = 0$, and number the vertices so that $V = \{1, \dots, n\}$ and

$$z_1 \geq z_2 \geq \cdots \geq z_n.$$

Then $S_i = \{1, \dots, i\}$ for $0 \leq i \leq n$. For $1 \leq i \leq n-1$, let

$$\Phi_i = \frac{\mathrm{Cap}(S_i, \bar{S}_i)}{\min\{\mathrm{Mass}(S_i), \mathrm{Mass}(\bar{S}_i)\}},$$

let $\Phi^* = \min_{1 \leq i \leq n-1} \Phi_i$, and let $R = z^\top Lz / z^\top Mz$. We must show that $\Phi^* \leq \sqrt{2\delta R}$. Our strategy is to modify z and G in two steps, neither of which increases the Rayleigh quotient or changes any Φ_i, and then to sum by parts.

Let

$$\beta = \min\{i \in \{1, \dots, n\} \mid \mathrm{Mass}(\bar{S}_i) \leq \mathrm{Mass}(S_i)\}.$$

This is well defined, since $i = n$ qualifies. As i grows, $\mathrm{Mass}(S_i)$ increases and $\mathrm{Mass}(\bar{S}_i)$ decreases. Hence the minimum in Φ_i is $\mathrm{Mass}(S_i)$ for $i < \beta$ and $\mathrm{Mass}(\bar{S}_i)$ for $i \geq \beta$.

The two modifications of z and G are:

1) shifting z so that $z_\beta = 0$;

2) pinching at β every edge (i, j) with $i < \beta < j$; these include all edges whose endpoints have shifted values of opposite signs.

Step 1: shifting z. Let $y = z - z_\beta \mathbf{1}$. Then $y_1 \geq \cdots \geq y_\beta = 0 \geq y_{\beta+1} \geq \cdots \geq y_n$, and y has the same order and the same threshold cuts as z.

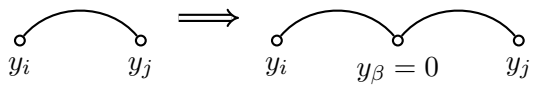
Claim 12.21.

$$\frac{y^\top Ly}{y^\top My} \leq \frac{z^\top Lz}{z^\top Mz}. \tag{a}$$

Proof. Since $L\mathbf{1} = 0$, the numerator does not change: $y^\top Ly = z^\top Lz$. Since $z^\top M\mathbf{1} = 0$ and M has positive diagonal,

$$y^\top My = z^\top Mz - 2z_\beta\, z^\top M\mathbf{1} + z_\beta^2\, \mathbf{1}^\top M\mathbf{1} = z^\top Mz + z_\beta^2\, \mathbf{1}^\top M\mathbf{1} \geq z^\top Mz > 0. \qquad \square$$

Step 2: pinching edges at β. For each edge (i, j) with $i < \beta < j$, so that $y_i \geq 0 \geq y_j$, we pinch the edge at the vertex β, where $y_\beta = 0$: we replace it by the two edges (i, β) and (β, j), each of weight w_{ij} (adding weights if these edges already exist).



Let G' be the resulting graph, with weights w', Laplacian L', degree matrix D' and sum-Laplacian $P' = 2D' - L'$. The masses are unchanged. In G', no edge joins a vertex $i < \beta$ to a vertex $j > \beta$.

Claim 12.22.

$$y^\top L'y \leq y^\top Ly. \tag{b}$$

Proof. Pinching replaces each term $w_{ij}(y_i - y_j)^2$ with $w_{ij}((y_i - 0)^2 + (0 - y_j)^2)$, and $y_i y_j \le 0$, so

$$(y_i - y_j)^2 = y_i^2 - 2y_i y_j + y_j^2 \ge y_i^2 + y_j^2. \qquad \square$$

Claim 12.23. *Pinching changes no Φ_k:* $\mathrm{Cap}_{G'}(S_k, \bar{S}_k) = \mathrm{Cap}_G(S_k, \bar{S}_k)$ *for* $1 \le k \le n-1$.

Proof. Let (i, j) be a pinched edge, $i < \beta < j$. If $k < \beta$, then $j, \beta \notin S_k$, so the edge (i, j) crosses S_k exactly when $i \le k$, and so does (i, β), while (β, j) never does. If $k \ge \beta$, then $i, \beta \in S_k$, so (i, j) crosses S_k exactly when $j > k$, and so does (β, j), while (i, β) never does. In both cases the crossing weight is unchanged. Since the masses are also unchanged, so is every Φ_k. $\square$

Pinching can increase only the degree of β; thus $D' \ge D$ entrywise, with $D'_{ii} = D_{ii}$ for $i \ne \beta$. Since $y_\beta = 0$, it follows that $y^\top D' y = y^\top D y$. For the sum-Laplacian of G', using $y^\top L' y \ge 0$,

$$\begin{aligned} \frac{y^\top P' y}{y^\top M y} &= \frac{y^\top (2D' - L')y}{y^\top M y} \le \frac{2y^\top D' y}{y^\top M y} = 2\,\frac{y^\top D y}{y^\top M y} \\ &= 2\,\frac{\sum_i d_i y_i^2}{\sum_i m_i y_i^2} \le 2 \max_i \Big(\frac{d_i}{m_i}\Big) = 2\delta, \end{aligned} \tag{c}$$

where the last step uses $d_i \le \delta m_i$ for every i.

Combining. By (c), then (a) and (b) (which give $y^\top L' y \le y^\top L y \le R\, y^\top M y$), and then Lemma 12.20 for G',

$$\begin{aligned} 2\delta R\,(y^\top M y)^2 &\overset{\text{(c)}}{\ge} (y^\top P' y)\, R\, (y^\top M y) \\ &\overset{\text{(a),(b)}}{\ge} (y^\top P' y)(y^\top L' y) \ge \Big(\sum_{(i,j)\in E'} w'_{ij} |y_i^2 - y_j^2| \Big)^2. \end{aligned}$$

Every edge of G' has both endpoints in $\{1, \dots, \beta\}$ or both in $\{\beta, \dots, n\}$, and not both. For an edge with $i < j \le \beta$ we have $y_i \ge y_j \ge 0$, so $|y_i^2 - y_j^2| = y_i^2 - y_j^2$; for an edge with $\beta \le i < j$ we have $0 \ge y_i \ge y_j$, so $|y_i^2 - y_j^2| = y_j^2 - y_i^2$. Taking square roots therefore gives

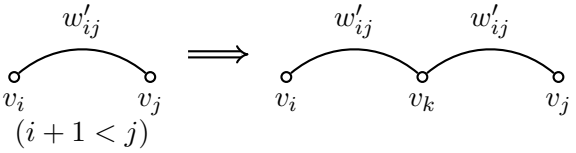
$$\sqrt{2\delta R}\,(y^\top M y) \ge \sum_{\substack{(i,j)\in E' \\ i<j\le\beta}} w'_{ij}(y_i^2 - y_j^2) + \sum_{\substack{(i,j)\in E' \\ \beta\le i<j}} w'_{ij}(y_j^2 - y_i^2). \tag{d}$$

Claim 12.24. *Without loss of generality G' is a weighted path graph, since the sums in* (d) *telescope. Precisely, let $w_k = \mathrm{Cap}(S_k, \bar{S}_k)$ for $1 \le k \le n-1$. Then*

$$\sum_{\substack{(i,j)\in E' \\ i<j\le\beta}} w'_{ij}(y_i^2 - y_j^2) = \sum_{k=1}^{\beta-1} w_k (y_k^2 - y_{k+1}^2),$$

$$\sum_{\substack{(i,j)\in E' \\ \beta\le i<j}} w'_{ij}(y_j^2 - y_i^2) = \sum_{k=\beta}^{n-1} w_k (y_{k+1}^2 - y_k^2).$$

In words: replacing each edge (i, j) with $i + 1 < j$ by the path $i, i+1, \dots, j$, each of whose edges has weight w'_{ij}, turns G' into a path in which the edge $(k, k+1)$ has weight w_k:



Proof. For an edge with $i < j \le \beta$, write $y_i^2 - y_j^2 = \sum_{k=i}^{j-1}(y_k^2 - y_{k+1}^2)$. Exchanging the order of summation, the coefficient of $y_k^2 - y_{k+1}^2$, for $1 \le k \le \beta - 1$, is the total weight of the edges of G' with $i \le k < j \le \beta$. Since no edge of G' joins a vertex $i < \beta$ to a vertex $j > \beta$, these are exactly the edges of G' that cross S_k, so the coefficient is $\mathrm{Cap}_{G'}(S_k, \bar{S}_k) = w_k$ by Claim 12.23. The second identity is proved in the same way, with $y_j^2 - y_i^2 = \sum_{k=i}^{j-1}(y_{k+1}^2 - y_k^2)$ for $\beta \le k \le n-1$. □

The main inequality. Recall that $y_1^2 \ge y_2^2 \ge \cdots \ge y_\beta^2 = 0$, so every difference $y_k^2 - y_{k+1}^2$ with $k < \beta$ is nonnegative, and that $\mathrm{Mass}(S_0) = 0$, since $S_0 = \emptyset$. For $1 \le k \le \beta - 1$, the minimum in Φ_k is $\mathrm{Mass}(S_k)$, so $w_k = \Phi_k \,\mathrm{Mass}(S_k)$ and $\Phi_k \ge \Phi^*$. Hence

$$\begin{aligned}
\sum_{k=1}^{\beta-1} w_k(y_k^2 - y_{k+1}^2) &= \sum_{k=1}^{\beta-1} \Phi_k \,\mathrm{Mass}(S_k)(y_k^2 - y_{k+1}^2) \\
&\ge \Phi^* \sum_{k=1}^{\beta-1} \mathrm{Mass}(S_k)(y_k^2 - y_{k+1}^2) \\
&= \Phi^* \sum_{k=1}^{\beta-1} \big(\mathrm{Mass}(S_k) - \mathrm{Mass}(S_{k-1})\big) y_k^2 \\
&= \Phi^* \sum_{k=1}^{\beta-1} m_k y_k^2 = \Phi^* \, y_{\text{left}}^\top M y_{\text{left}},
\end{aligned}$$

where y_{left} agrees with y on $\{1, \ldots, \beta\}$ and is 0 elsewhere. The third line is summation by parts: the two sums differ by $\mathrm{Mass}(S_0)\, y_1^2 - \mathrm{Mass}(S_{\beta-1})\, y_\beta^2$, which is 0. In the same way, for $\beta \le k \le n-1$ we have $w_k = \Phi_k \,\mathrm{Mass}(\bar{S}_k)$ and $y_{k+1}^2 - y_k^2 \ge 0$, and summation by parts, with $\mathrm{Mass}(\bar{S}_{k-1}) - \mathrm{Mass}(\bar{S}_k) = m_k$ and $\mathrm{Mass}(\bar{S}_n) = 0$, gives

$$\sum_{k=\beta}^{n-1} w_k(y_{k+1}^2 - y_k^2) \ge \Phi^* \sum_{k=\beta+1}^{n} m_k y_k^2 = \Phi^* \, y_{\text{right}}^\top M y_{\text{right}},$$

where y_{right} agrees with y on $\{\beta, \ldots, n\}$ and is 0 elsewhere. Since $y_\beta = 0$, $y^\top M y = y_{\text{left}}^\top M y_{\text{left}} + y_{\text{right}}^\top M y_{\text{right}}$. (If $\beta = 1$ or $\beta = n$, one of the two sums is empty and the corresponding part of y is zero.)

Therefore, by (d) and Claim 12.24,

$$\sqrt{2\delta R}\; (y^\top M y) \ge \Phi^* (y_{\text{left}}^\top M y_{\text{left}} + y_{\text{right}}^\top M y_{\text{right}}) = \Phi^* (y^\top M y).$$

Since $y^\top M y > 0$, we conclude that

$$\Phi^* \le \sqrt{2\delta \, \frac{z^\top L z}{z^\top M z}},$$

so the best threshold cut of z satisfies the theorem. □

13 Direct Linear Solvers, Nested Dissection for Planar Graphs

Lecture of February 17, 2020. Based on Gary L. Miller's handwritten notes. Zhen Zhou's scribe notes for this lecture were used as a reading aid.

This lecture studies direct methods, which solve a sparse linear system $Mx = b$ exactly. Gaussian elimination factors M into triangular matrices, and for symmetric positive definite matrices every pivot order works, so the order can be chosen to keep the factors sparse. We translate pivoting into an operation on the graph of M: pivoting a vertex turns its neighborhood into a clique, and the new edges are the fill. We compare pivot orders on trees, interval graphs and chordal graphs, and show that the graphs with fill added are always chordal. For the square mesh, the column-by-column order has fill $\Omega(n^{3/2})$, while nested dissection achieves fill $O(n \log n)$ and work $O(n^{3/2})$; with block elimination and a fast dense solver, the work drops to $O(n^{\alpha/2})$.

13.1 Direct linear system solvers

Throughout this lecture, M is an $n \times n$ matrix over a field, and $w(M)$ denotes the number of nonzero entries of M. We are interested in sparse matrices, for which $w(M)$ is much smaller than n^2. Given M and a vector b, a *direct method* solves $Mx = b$ exactly: it performs a finite number of arithmetic operations and returns the exact solution x.

The simplest case is a triangular system. A matrix L is *lower triangular* if $l_{ij} = 0$ whenever $j > i$, that is,

$$L = \begin{pmatrix} * & & 0 \\ & \ddots & \\ * & & * \end{pmatrix},$$

and a matrix is *upper triangular* if its transpose is lower triangular.

Claim 13.1. *Let L be an $n \times n$ lower triangular matrix with nonzero diagonal entries. Then for every b, the system $Lx = b$ has a unique solution, and it can be computed with $O(w(L))$ arithmetic operations.*

Proof. The first equation reads $l_{11}x_1 = b_1$, so $x_1 = b_1/l_{11}$. Substituting this value into the other equations leaves the system

$$\begin{pmatrix} l_{22} & & 0 \\ \vdots & \ddots & \\ l_{n2} & \cdots & l_{nn} \end{pmatrix} \begin{pmatrix} x_2 \\ \vdots \\ x_n \end{pmatrix} = b', \qquad b'_i = b_i - l_{i1}x_1 \quad (2 \le i \le n),$$

which is again lower triangular with nonzero diagonal entries. Computing x_1 and b' takes one division, and one multiplication and one subtraction for each nonzero entry l_{i1} with $i \ge 2$; that is, $O(1)$ operations per nonzero entry of the first column of L. By induction on n, the remaining unknowns are determined uniquely, and the total number of operations is $O(w(L))$. □

The same argument, solving for x_n first, applies to upper triangular matrices with nonzero diagonal entries. We write U for an upper triangular matrix.

Suppose that $M = LU$, where L is lower triangular and U is upper triangular, both with nonzero diagonal entries. Then we can solve $Mx = b$ with $O(w(L) + w(U))$ operations, as follows.

Algorithm 13.2. Given L, U and b with $M = LU$:

1) solve $Ly = b$;

2) solve $Ux = y$.

Then $Mx = LUx = Ly = b$, and by Claim 13.1 each step takes $O(w(L))$ or $O(w(U))$ operations.

13.2 Computing an LU decomposition

An LU decomposition is computed by Gaussian elimination; for symmetric positive definite matrices, its symmetric form is the Cholesky decomposition. Gaussian elimination may require pivoting, that is, reordering the rows, as the following example shows.

Example 13.3. Let

$$M = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \end{pmatrix}.$$

Subtracting row 1 from rows 2 and 3 is multiplication on the left by a lower triangular matrix:

$$\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

The next step should clear the entry in position $(3, 2)$ by subtracting a multiple of row 2 from row 3. This is impossible, because the entry in position $(2, 2)$, the pivot, is 0. So we swap rows 2 and 3 of M and start over:

$$\begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix},$$

which is already upper triangular.

13.2.1 Elimination in general

We assume that the rows of M have been ordered in advance ($M \leftarrow PM$ for a permutation matrix P) so that elimination never meets a zero pivot. Let e_i denote the ith standard basis vector. For $1 \le i \le n-1$, let

$$L_i = I + \ell_i e_i^\top = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & * & \ddots & \\ & & \vdots & & \\ & & * & & 1 \end{pmatrix},$$

the identity except for the starred entries in column i below the diagonal; the vector ℓ_i holds these entries, so its first i entries are 0. Step i of elimination multiplies on the left by L_i, with

the entries of ℓ_i chosen to clear column i below the diagonal: if $M^{(i)} = L_{i-1}\cdots L_1 M$, then $(\ell_i)_j = -M^{(i)}_{ji}/M^{(i)}_{ii}$ for $j > i$. After $n-1$ steps,

$$L_{n-1}\cdots L_2 L_1 M = U$$

is upper triangular.

Let $L \coloneqq L_{n-1}\cdots L_1$, so that $LM = U$. As a product of lower triangular matrices, L is lower triangular, but its entries have no simple closed form. Its inverse

$$L^{-1} = L_1^{-1}\cdots L_{n-1}^{-1}$$

is much simpler.

Claim 13.4. *If*

$$L_1 = \begin{pmatrix} 1 & & & \\ a_1 & 1 & & \\ \vdots & & \ddots & \\ a_{n-1} & & & 1 \end{pmatrix}, \quad \textit{then} \quad L_1^{-1} = \begin{pmatrix} 1 & & & 0 \\ -a_1 & 1 & & \\ \vdots & & \ddots & \\ -a_{n-1} & & & 1 \end{pmatrix}.$$

If moreover

$$L_2 = \begin{pmatrix} 1 & & & & 0 \\ 0 & 1 & & & \\ \vdots & b_1 & 1 & & \\ \vdots & \vdots & 0 & \ddots & \\ 0 & b_{n-2} & 0 & & 1 \end{pmatrix}, \quad \textit{then} \quad L_1^{-1}L_2^{-1} = \begin{pmatrix} 1 & & & & 0 \\ -a_1 & 1 & & & \\ \vdots & -b_1 & 1 & & \\ \vdots & \vdots & 0 & \ddots & \\ -a_{n-1} & -b_{n-2} & 0 & & 1 \end{pmatrix}.$$

In general, $L_i^{-1} = I - \ell_i e_i^\top$ *and* $L_1^{-1}\cdots L_{n-1}^{-1} = I - \sum_{i=1}^{n-1} \ell_i e_i^\top$.

Proof. For $i \le j$, the product $e_i^\top \ell_j$ is the ith entry of ℓ_j, which is 0. Hence $(I + \ell_i e_i^\top)(I - \ell_i e_i^\top) = I - \ell_i(e_i^\top \ell_i)e_i^\top = I$. Expanding the product $(I - \ell_1 e_1^\top)\cdots(I - \ell_{n-1}e_{n-1}^\top)$, every term with two or more factors $\ell_i e_i^\top$ contains a factor $e_i^\top \ell_j$ with $i < j$, so it vanishes, and only $I - \sum_i \ell_i e_i^\top$ remains. The two displayed matrices are the cases L_1^{-1} and $L_1^{-1}L_2^{-1}$. □

Claim 13.5. $L^{-1} = nI - L_1 - \cdots - L_{n-1}$.

Proof. We have $\sum_{i=1}^{n-1} L_i = (n-1)I + \sum_i \ell_i e_i^\top$, so $nI - \sum_i L_i = I - \sum_i \ell_i e_i^\top$, which equals L^{-1} by Claim 13.4. □

Hence

$$M = L^{-1}U,$$

where L^{-1} is lower triangular with unit diagonal, and its entries below the diagonal are the negated entries of the ℓ_i. This is the LU decomposition computed by Gaussian elimination.

13.3 Symmetric positive definite matrices

For general matrices the order of the pivots is constrained, as the example shows. For the important class of symmetric positive definite matrices we may pick any pivot order.

Definition 13.6. A real $n \times n$ matrix M is *symmetric positive definite* (SPD) if

1) M is symmetric: $M^\top = M$, and
2) M is positive definite: $x^\top M x > 0$ for every $x \neq 0$; equivalently, all eigenvalues of M are positive.

For a symmetric matrix we pivot symmetrically: choosing the diagonal entry m_{ii} as the next pivot means eliminating the variable x_i next, which reorders both the rows and the columns ($M \leftarrow PMP^\top$). This keeps M SPD.

Homework 13.7. Let M be SPD. Show that every diagonal entry of M is positive, and that this remains true under Gaussian elimination: after the first elimination step, the remaining $(n-1) \times (n-1)$ block (the matrix M_1 of Section 13.4) is again SPD. Conclude that, for every pivot order, Gaussian elimination on M never meets a zero pivot.

So for SPD matrices we may choose the pivot order freely. Our goal is to choose it so that the triangular factor L has as few nonzero entries as possible, that is, to minimize $w(L)$. The pattern of nonzero entries is described by a graph.

Definition 13.8. Let M be a symmetric $n \times n$ matrix. The *graph* of M is $\mathrm{Graph}(M) = G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ and, for $i \neq j$, $(v_i, v_j) \in E$ if $M_{ij} \neq 0$.

Since M is symmetric, G is undirected.

13.4 Schur complement

Let M be symmetric with $m_{11} = a \neq 0$, and write

$$M = \begin{pmatrix} a & c^\top \\ c & C \end{pmatrix}, \qquad L_1 = \begin{pmatrix} 1 & 0 \\ -c/a & I \end{pmatrix},$$

where $c \in \mathbb{R}^{n-1}$ and C is $(n-1) \times (n-1)$. The first elimination step gives

$$L_1 M = \left(\begin{array}{c|ccc} a & * & \cdots & * \\ \hline 0 & & & \\ \vdots & & M_1 & \\ 0 & & & \end{array}\right), \qquad M_1 = C - \frac{1}{a}\, c\, c^\top.$$

Multiplying on the right by $L_1^\top$ subtracts multiples of the first column from the other columns. This clears the first row and leaves the lower right block unchanged, because the first column of $L_1 M$ is zero below a:

$$L_1 M L_1^\top = \left(\begin{array}{c|ccc} a & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & M_1 & \\ 0 & & & \end{array}\right).$$

Since

$$(L_1 M L_1^\top)^\top = (L_1^\top)^\top M^\top L_1^\top = L_1 M L_1^\top,$$

the matrix $L_1 M L_1^\top$ is symmetric, and therefore so is M_1.

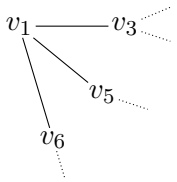
Definition 13.9. The matrix $M_1 = C - \frac{1}{a}\, c\, c^\top$ is the *Schur complement* of the entry $a = m_{11}$ in M.

We now describe $\mathrm{Graph}(M_1)$ in terms of $G = \mathrm{Graph}(M)$. Index the rows and columns of M_1 by $2, \ldots, n$. For $2 \le j < k \le n$,

$$(M_1)_{jk} = m_{jk} - \frac{m_{j1}\, m_{1k}}{m_{11}}.$$

Hence $(M_1)_{jk}$ can be nonzero only if $(v_j, v_k) \in E$, or if both (v_j, v_1) and (v_1, v_k) are edges.

Example 13.10. Suppose that v_1 is adjacent to exactly v_3, v_5 and v_6:



The first row and column of M have nonzero entries exactly in positions 1, 3, 5 and 6; the other entries of M are arbitrary. The first pivot clears column 1 below the diagonal, and otherwise it changes only the entries of $L_1 M$ that lie in rows 3, 5, 6 and columns 3, 5, 6:

$$M = \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & & \end{array} \\ \begin{pmatrix} * & 0 & * & 0 & * & * & 0 & \cdots \\ 0 & & & & & & & \\ * & & & & & & & \\ 0 & & & & & & & \\ * & & & & & & & \\ * & & & & & & & \\ 0 & & & & & & & \\ \vdots & & & & & & & \end{pmatrix} \end{array}, \qquad L_1 M = \begin{array}{c} \\ 3 \\ \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} * & 0 & * & 0 & * & * \\ 0 & & & & & \\ 0 & & * & & \boxed{*} & \boxed{*} \\ 0 & & & & & \\ 0 & & * & & * & * \\ 0 & & * & & * & * \end{pmatrix} \end{array}.$$

The boxed entries may be nonzero even if $m_{35} = m_{36} = 0$; they are possibly new edges of $\mathrm{Graph}(M_1)$. The edge (v_3, v_5) comes from the edges (v_3, v_1) and (v_1, v_5), and the edge (v_3, v_6) from (v_3, v_1) and (v_1, v_6).

13.5 Graph theoretic view of pivoting

The computation above says that a pivot acts on the graph as follows.

Definition 13.11. Let G be a graph and v_i a vertex of G. The operation $\mathrm{Pivot}(v_i)$ is:

1) make a clique of the set $\mathrm{Neig}(v_i)$ of neighbors of v_i, by adding every missing edge between two of them;

2) remove v_i and its edges.

Fill(v_i) denotes the set of new edges added in step 1. A *pivot strategy* is a pivot order $v_1, \dots, v_n$ of the vertices: we apply Pivot(v_1) to G, then Pivot(v_2) to the result, and so on. We write $\hat{G}_i$ for the graph obtained after pivoting $v_1, \dots, v_i$ ($\hat{G}_0 = G$), and d_i for the degree of v_i in $\hat{G}_{i-1}$, its degree at pivot time. The *fill* of the strategy is the set F of all edges added by its pivots, and $G + F$ is the *filled graph*. The *cost* of Pivot(v_i) is $O(d_i^2)$, since it updates the entries indexed by pairs of neighbors of v_i; in the estimates below we count it as $d_i^2/2$. The *work* of a pivot strategy is the total cost of its pivots.

Adding edges to a graph can only enlarge the filled graph and increase the work. The number of fill edges itself can drop, because an added edge may be one that a pivot would otherwise have created.

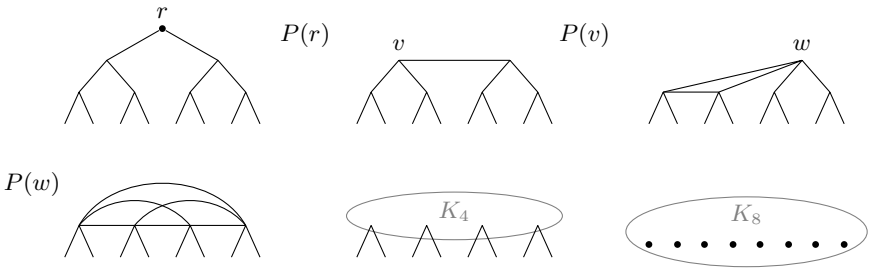
Lemma 13.12. *Let $G \subseteq G'$ be graphs on the same vertex set, and fix a pivot order $v_1, \dots, v_n$. Then $\hat{G}_i \subseteq \hat{G}'_i$ for every i. Consequently $d_i \le d'_i$ for every i, the work for G is at most the work for G', and $G + F \subseteq G' + F'$.*

Proof. By induction on i, the case $i = 0$ being the hypothesis. If $\hat{G}_{i-1} \subseteq \hat{G}'_{i-1}$, then the neighbors of v_i in $\hat{G}_{i-1}$ are neighbors of v_i in $\hat{G}'_{i-1}$, so the clique added on them in $\hat{G}_{i-1}$ is contained in the clique added in $\hat{G}'_{i-1}$; deleting v_i from both graphs preserves the inclusion. The filled graph $G + F$ is the union of the graphs $\hat{G}_0, \hat{G}_1, \dots$, so it is contained in $G' + F'$. □

By the formula for M_1, if M is SPD with graph G, then Graph(M_1) is a subgraph of the graph obtained from G by Pivot(v_1), and equality holds unless entries cancel exactly. Repeating this (with Lemma 13.12 at each step), the nonzero entries that Gaussian elimination with pivot order $v_1, \dots, v_n$ creates lie in positions that are edges of $G + F$. In particular, column i of the factor has at most d_i nonzero entries below the diagonal, and $\sum_i d_i = |E| + |F|$, since each edge of $G + F$ is counted once, at the pivot of its endpoint that comes first. So minimizing $w(L)$ amounts to minimizing the fill.

Example 13.13 (balanced binary tree)**.** Let G be the balanced binary tree with levels 0 (the root r) to h, so that G has $n = 2^{h+1} - 1$ vertices, of which $(n + 1)/2$ are leaves. In the figures, $P(v)$ stands for Pivot(v).

Pivot strategy 1: pivot from the root to the leaves, level by level.



As the panels show, when the vertices of levels $0, \dots, k-1$ have been pivoted, the vertices of level k form a clique, for every $1 \le k \le h$. To see this, suppose that level k is a clique U and that the vertices below level k still carry only their tree edges, and pivot the vertices $u_1, u_2, \dots$ of U in turn. By induction on j, after $u_1, \dots, u_j$ are pivoted, the remaining vertices of U together with the children of $u_1, \dots, u_j$ form a clique: the neighbors of u_{j+1} are the members of this clique and its own two children, and pivoting u_{j+1} joins all of them. When U is used up, the clique

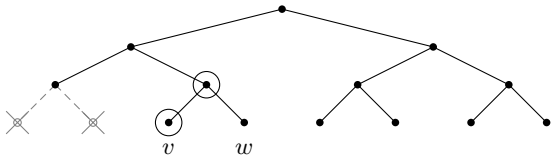
is level $k+1$. In particular, after all internal vertices are pivoted, the $(n+1)/2$ leaves form a clique $K_{(n+1)/2}$. No two leaves are adjacent in G, so

$$|F| \geq \binom{(n+1)/2}{2} = \frac{n^2-1}{8} = \Omega(n^2).$$

The work to pivot the clique on the leaves alone is $\Omega(n^3)$: the jth leaf to be pivoted still has $(n+1)/2 - j$ neighbors, so the work is at least $\sum_{d=0}^{(n-1)/2} d^2/2 = \Omega(n^3)$.

Pivot strategy 2: pivot from the leaves to the root, level by level, starting at level h. All pivots are on vertices of degree at most one: when a vertex is pivoted, its children have already been pivoted, so its only remaining neighbor is its parent (the root has none). A set of at most one vertex is already a clique. Hence

1) there is no fill, and
2) each pivot costs $O(1)$, so the work is $O(n)$.



In the figure, the two crossed-out leaves have been pivoted. The leaf v has degree one, its only neighbor being its parent, so pivoting it creates no fill; after v and its sibling w are pivoted, their parent is a leaf.

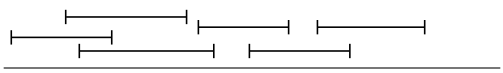
13.6 Perfect elimination

Definition 13.14. A *perfect elimination order* of G is a pivot order with *no* fill, that is, with $F = \emptyset$.

Trees have perfect elimination orders. Strategy 2 of Example 13.13 is one for the balanced binary tree; for an arbitrary tree, repeatedly pivot a vertex of degree at most one, which exists in every nonempty forest, and whose removal leaves a forest.

Definition 13.15. A graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$ is an *interval graph* if there are closed intervals $I_1, \ldots, I_n \subseteq \mathbb{R}$ such that

1) vertex v_i corresponds to the interval I_i, and
2) for $i \neq j$, $(v_i, v_j) \in E$ if and only if $I_i \cap I_j \neq \emptyset$.



Proposition 13.16. *Every interval graph has a perfect elimination order: pivot the vertices in order of increasing right endpoints of their intervals.*

Proof. Let $I = [l, r]$ be the interval with the leftmost right endpoint. Then $\mathrm{Neig}(I)$ is a clique. Indeed, every neighbor $J = [l_J, r_J]$ of I meets I, so $l_J \leq r$, and $r \leq r_J$ by the choice of I; hence every neighbor of I contains the point r, and any two of them intersect. So pivoting I creates no fill, and the remaining graph is the interval graph of the remaining intervals. By induction on the number of intervals, the stated order has no fill. □

13.7 Chordal graphs

A *chord* of a cycle is an edge that joins two vertices of the cycle that are not consecutive on it.

Definition 13.17. A graph G is *chordal* if every cycle of length at least 4 in G has a chord.

Theorem 13.18. *Let G be a graph, and let F be the fill of Gaussian elimination on G with some pivot order. Then $G + F$ is chordal.*

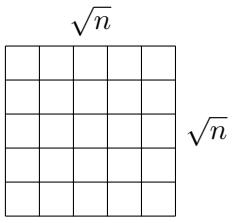
Proof. Each edge of $G + F$ is an edge of G or is created by a pivot made while both of its endpoints are unpivoted; in both cases it is present in the current graph until one of its endpoints is pivoted. Let C be a cycle of length at least 4 in $G + F$, and let v be the vertex of C that is pivoted first. When v is pivoted, no other vertex of C has been pivoted, so the two neighbors u and u' of v on C are neighbors of v in the current graph. Pivoting v makes u and u' adjacent, so (u, u') is an edge of $G+F$. Since C has length at least 4, the vertices u and u' are not consecutive on C, so (u, u') is a chord of C. □

Definition 13.19. A graph $H \supseteq G$ on the same vertex set as G is a *chordal completion* of G if H is chordal.

By Theorem 13.18, every pivot strategy produces a chordal completion $G + F$ of G, which has $|F|$ more edges than G. Thus an optimal pivot strategy amounts to finding a minimum chordal completion. For the converse direction we use a fact that we do not prove: every chordal graph has a perfect elimination order [FG65, Ros70]. If H is a chordal completion of G and we pivot G in a perfect elimination order of H, then by Lemma 13.12 every fill edge of G is an edge of H. Hence the minimum fill over all pivot strategies equals the minimum of $|E(H)| - |E(G)|$ over all chordal completions H of G.

13.8 Pivot strategies for the square mesh

For an integer $s \geq 1$, the $s \times s$ *square mesh* Mesh_s is the graph with vertex set $\{0, 1, \ldots, s\}^2$ in which two vertices are adjacent when they agree in one coordinate and differ by 1 in the other. We write $n = s^2$, so the mesh has $\sqrt{n}$ edges on each side and $(\sqrt{n} + 1)^2 = \Theta(n)$ vertices. For $0 \leq x \leq s$, the *column* C_x is the set of vertices (x, y) with $0 \leq y \leq s$.

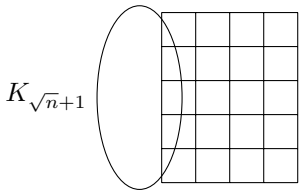


Our input is a system $Ax = b$ with A SPD and $\mathrm{Graph}(A) = \mathrm{Mesh}_s$. We want to show that the best pivot strategies for the mesh satisfy

$$\mathrm{Fill}(\mathrm{Mesh}) = \Theta(n \log n),$$
$$\mathrm{Work}(\mathrm{Mesh}) = \Theta(n^{3/2}).$$

We prove the upper bounds, by nested dissection (Theorem 13.21). We do not prove the lower bounds, which say that every pivot strategy for the mesh has fill $\Omega(n \log n)$ and work $\Omega(n^{3/2})$ [HMR73].

A first try is to pivot one column at a time: the columns $C_0, C_1, \ldots, C_s$ in turn, each from $y = 0$ up to $y = s$. After the first column, the second column has become a clique $K_{\sqrt{n}+1}$:



In fact each column will be a clique.

Claim 13.20. *In the column-by-column order, for every $1 \le x \le s$, the vertices of C_x form a clique when the pivoting of C_x begins. Moreover, for $1 \le x \le s-1$, every vertex of C_x has degree exactly $s+1$ at pivot time.*

Proof. We show by induction on x that when the pivoting of C_x begins, the current graph is the mesh restricted to $C_x \cup \cdots \cup C_s$, together with a clique on C_x if $x \ge 1$. This holds for $x = 0$. Suppose it holds for some $x \le s-1$. Set $A_y = \{(x,y), \ldots, (x,s)\} \cup \{(x+1,0), \ldots, (x+1, y-1)\}$ if $x \ge 1$, and $A_y = \{(0,y)\} \cup \{(1,0), \ldots, (1, y-1)\}$ if $x = 0$. We claim that when (x,y) is pivoted, A_y is a clique and the other edges are mesh edges. This holds for $y = 0$. The neighbors of (x,y) are then the other members of A_y, the vertex $(x+1, y)$, and, if $x = 0$, the vertex $(0, y+1)$. Pivoting (x,y) joins all of them, which makes A_{y+1} a clique. After (x,s) is pivoted, the clique is $A_{s+1} = C_{x+1}$, and the other edges are the mesh edges of $C_{x+1} \cup \cdots \cup C_s$. For $x \ge 1$, the vertex (x,y) has $|A_y| - 1 + 1 = s+1$ neighbors at pivot time. □

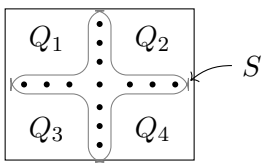
So each column C_x with $1 \le x \le s$ becomes a clique on $s+1$ vertices, of which only s pairs are mesh edges. Hence

$$|F| \ge s\Big(\binom{s+1}{2} - s\Big) = \frac{s^2(s-1)}{2} = \Omega(n^{3/2}),$$

and the $(s-1)(s+1)$ pivots in the columns $C_1, \ldots, C_{s-1}$ each cost $(s+1)^2/2$, so the work is at least $(s-1)(s+1)^3/2 = \Omega(n^2)$.

13.9 Nested dissection

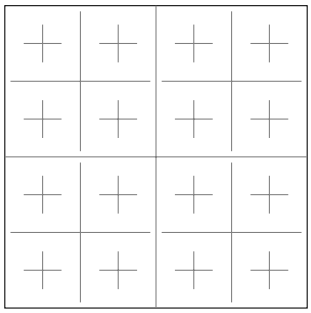
Nested dissection does much better. For our mesh with s even, let S be the cross formed by the middle row and the middle column, without the boundary of the mesh. The cross cuts the mesh into four quadrants $Q_1, \ldots, Q_4$, squares of side $s/2$:



Nested dissection (ND) pivots the vertices inside each quadrant recursively by nested dissection, and pivots the separator S last:

$$\text{Pivot(Mesh)} = \big(\text{Pivot}(Q_1), \text{Pivot}(Q_2), \text{Pivot}(Q_3), \text{Pivot}(Q_4), S\big).$$

Here Pivot(Q_j) pivots only the vertices inside the square Q_j, not those on its boundary; the vertices on the boundary of the mesh are pivoted at the very end. We obtain nested crosses, and we pivot the small ones first.



Concretely, let $s = 2^l$. Nested dissection proceeds in *rounds* $K = 1, 2, 4, \ldots, 2^{l-1}$. In round K, the mesh is divided into $(s/2K)^2 = n/4K^2$ blocks of size $2K \times 2K$ (with corners at the points whose coordinates are multiples of $2K$), and the *cross* of each block is pivoted: the vertices of its middle row and middle column that are not on the boundary of the block. The cross consists of the *horizontal part*, the $2K - 2$ vertices of the middle row other than the center, which form a left arm and a right arm, and the *vertical part*, the $2K - 1$ vertices of the middle column. Within a round, we first pivot all horizontal parts and then all vertical parts. After the last round, we pivot the $4s$ vertices on the boundary of the mesh. This is the recursive order above with the quadrants handled side by side rather than one after another. No edge joins vertices of two different quadrants until the separator between them is pivoted, so pivots in different quadrants do not affect each other, and the rearrangement changes neither the fill nor the work.

Our goal is to bound the fill and the work of nested dissection.

Theorem 13.21. *Let $s = 2^l$ and $n = s^2$. Nested dissection on* Mesh_s *has fill at most $\frac{17}{4}\, n \log_2 n + 10n$ and work $O(n^{3/2})$.*

The proof, in Section 13.11, counts fill by following the faces of the planar drawing of the mesh.

13.10 A view of fill for planar graphs

Let G be a triangulated planar graph: G is drawn in the plane without crossings, and every face, including the outer one, is bounded by three edges. For a face f of a drawing, $V(f)$ denotes the set of vertices on the boundary of f. Let $v_1, \ldots, v_n$ be a pivot strategy, and let

$$G_i \coloneqq G \setminus \{v_1, \ldots, v_i\}$$

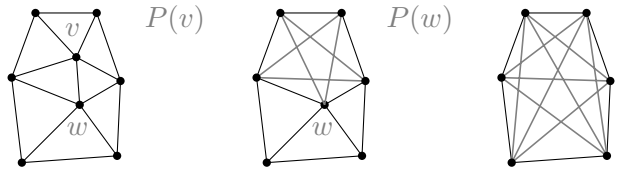
be the drawing with $v_1, \ldots, v_i$ and their edges deleted ($G_0 = G$). Removing v_i from G_{i-1} merges v_i, its edges and the faces around v_i into one new face F of G_i, and $V(F) = \bigcup_f V(f) \setminus \{v_i\}$, the union over the faces f of G_{i-1} whose boundary contains v_i.

Lemma 13.22. *For every i, two vertices of G_i are adjacent in $\hat{G}_i$ if and only if they lie on a common face of G_i. Equivalently,* Pivot(v_i) *acts as follows, given G_{i-1}:*

1) remove v_i from G_{i-1}, forming a new face F;

2) form a clique out of the vertices on F.

Proof. By induction on i. For $i = 0$, every edge of G lies on the boundary of some face, and the three vertices of a face are pairwise adjacent. Suppose the statement holds for $i - 1$. Then the neighbors of v_i in $\hat{G}_{i-1}$ are the vertices other than v_i on the faces of G_{i-1} at v_i, that is, the vertices of the new face F. So Pivot(v_i) makes $V(F)$ a clique and deletes v_i. The faces of G_i are F and the faces of G_{i-1} that do not contain v_i. Two vertices of G_i are adjacent in $\hat{G}_i$ if and only if they were adjacent in $\hat{G}_{i-1}$ or both lie on F. By the induction hypothesis, the first case means that they share a face f of G_{i-1}; if f contains v_i, both vertices lie on F, and otherwise f is a face of G_i. So they are adjacent in $\hat{G}_i$ if and only if they share a face of G_i. □



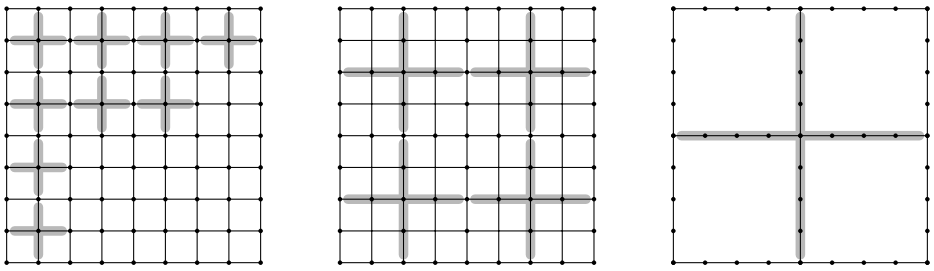
In the figure, pivoting v merges the five triangles at v into one face, whose six vertices become a clique (the gray edges); pivoting w then merges this face with the triangles at w.

Remark 13.23. The proof uses only that the vertices of every face of the drawing of G are pairwise adjacent. So for any graph G drawn in the plane without crossings, Lemma 13.22 holds for the graph $G^{\triangle}$ obtained from G by joining every two vertices that lie on a common face (pivoted in the same order, with G_i still the drawing of G with $v_1, \ldots, v_i$ deleted), although $G^{\triangle}$ itself need not be planar.

13.11 Fill and work for nested dissection

13.11.1 Pivoting out crosses for the square mesh

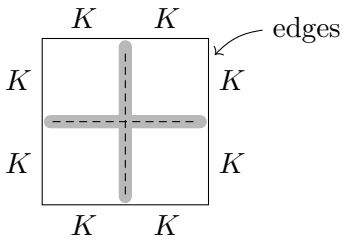
The rounds of nested dissection pivot out crosses of growing size:



Pivot 1 by 1 crosses | Pivot 2 by 2 crosses | Pivot 4 by 4 crosses

The panels show Mesh_8, with some of the crosses of the current round shaded; in each panel, the dots are the vertices not yet pivoted when the round begins. In general, for $0 < z < s$ let $\nu(z)$ be the largest power of 2 that divides z. A vertex (x, y) with $0 < x, y < s$ is pivoted in round $K = \max\{\nu(x), \nu(y)\}$: in the vertical part of a cross if $\nu(x) = K$, and in a horizontal arm otherwise. Hence, when round K begins, the vertices not yet pivoted are exactly those with x or y a multiple of K. So the drawing G_i at that moment is a grid of $K \times K$ squares, and the interior of each square is a face with $4K$ boundary vertices.

13.11.2 Removing crosses of size $2K \times 2K$



Proof of Theorem 13.21. The mesh is not triangulated, so we first add edges. Let G' be the graph $\text{Mesh}_s^{\triangle}$ of Remark 13.23: the mesh with both diagonals added in every unit square, and with the $4s$ vertices on the boundary of the mesh (the vertices of the outer face) joined into a clique. We bound the fill and the work of nested dissection on G', using Lemma 13.22 through Remark 13.23, and then return to the mesh. Recall that Mesh_s is $\sqrt{n} \times \sqrt{n}$ edges, with $\sqrt{n} = s = 2^l$. Each cross is pivoted out in two steps:

1) pivot out the horizontal part;

2) pivot out the vertical part.

Fill in round K. We start with the fill for crosses of size $2K \times 2K$. The number of crosses is

$$\left(\frac{\sqrt{n}}{2K}\right)^2 = \frac{n}{4K^2}.$$

Initially, by the paragraph above, the faces inside a block are four $K \times K$ squares, each with $4K$ vertices. Suppose first that $K \geq 2$.

H) The first pivot v of the left arm of a block lies on exactly two faces, the two left squares of the block, which share the $K+1$ vertices of the left arm and its endpoints. Pivoting v forms a new face from these two old faces. Any two vertices of one square are already adjacent, so the new edges join one of the $3K-1$ vertices of the upper square that are not on the shared side to one of the $3K-1$ such vertices of the lower square: at most $(3K)^2$ new fill. The remaining pivots of the arm lie only on the merged face, which is already a clique, so they have no fill. The same holds for the right arm, and there are $2 \cdot n/4K^2 = n/2K^2$ arms. Thus

$$\text{Fill}_H \leq \left(\frac{n}{2K^2}\right)(3K)^2 = \frac{9}{2}\,n.$$

V) After the horizontal parts are pivoted, the faces inside a block are its left and right halves, two $K \times 2K$ rectangles with $6K$ vertices each, which share the $2K+1$ vertices of the middle column. The first vertical pivot merges them and creates at most $(4K-1)^2 \leq (4K)^2$ fill edges, each joining a vertex of the left half to a vertex of the right half. The rest have no fill. Thus

$$\text{Fill}_V \leq \left(\frac{n}{4K^2}\right)(4K)^2 = 4n.$$

For $K = 1$ the arms are empty, and the vertical part is the center of the block alone. It lies on the four unit squares of its block, which contain 8 further vertices; 12 of the $\binom{8}{2} = 28$ pairs of

them lie in a common square and are already adjacent, so the pivot creates at most $16 = (4K)^2$ fill edges, and again $\mathrm{Fill}_H + \mathrm{Fill}_V \le 4n$.

Thus the fill per round is at most $\frac{17}{2}\,n$. The number of rounds is $l = \log_2 \sqrt{n}$. When the boundary of the mesh is pivoted at the end, the whole interior of the mesh has been merged into one face, so the $4s$ boundary vertices already form a clique, and they create no fill. Hence the total fill of nested dissection on G' is at most

$$\frac{17}{2}\, n \log_2 \sqrt{n} = \frac{17}{4}\, n \log_2 n.$$

Work in round K. By Lemma 13.22, the degree of a vertex at pivot time is the number of other vertices on the new face it forms. A horizontal pivot forms a face inside two $K \times K$ squares, with at most $4K + 4K - (K+1) < 7K$ vertices, and a vertical pivot forms a face inside one block, with at most $6K + 6K - (2K+1) < 10K$ vertices. Each cross has fewer than $2K$ horizontal and fewer than $2K$ vertical pivots. Counting $(\#\text{crosses})(\#\text{pivots})\frac{(\text{face size})^2}{2}$, the horizontal pivots of round K cost at most

$$\Big(\frac{n}{4K^2}\Big)(2K)\Big(\frac{(7K)^2}{2}\Big) = \frac{49}{4}\, n \cdot K,$$

and the vertical pivots cost at most

$$\Big(\frac{n}{4K^2}\Big)(2K)\Big(\frac{(10K)^2}{2}\Big) = 25\, n \cdot K.$$

(For $K = 1$, each center has degree $8 < 10K$ at pivot time, so the bound holds as well.) With $K = 2^i$ for $0 \le i \le l-1$, the total work of the rounds is at most

$$\sum_{i=0}^{l-1} \frac{149}{4}\, n\, 2^i < \frac{149}{4}\, n\, 2^l = \frac{149}{4}\, n^{3/2}.$$

The $4s$ boundary pivots at the end have degree less than $4s$, so they cost less than $(4s)^3/6 = \frac{32}{3}\, n^{3/2}$ in total. So the work of nested dissection on G' is less than $49\, n^{3/2}$.

Back to the mesh. Since $\mathrm{Mesh}_s \subseteq G'$ on the same vertex set, Lemma 13.12 shows that the work on the mesh is at most the work on G', which is $O(n^{3/2})$, and that the filled graph of the mesh is contained in that of G'. Hence the fill of the mesh is at most the fill of G' plus the number of edges of G' that are not mesh edges. These are the $2s^2 = 2n$ diagonals and fewer than $(4s)^2/2 = 8n$ edges among the boundary vertices, so the fill of the mesh is at most $\frac{17}{4}\, n \log_2 n + 10n$. □

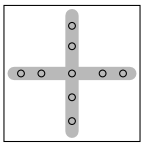
13.12 Block Gaussian elimination

Pivots can also be done a block at a time. For a symmetric matrix with blocks A, B, C, where A is invertible,

$$\begin{pmatrix} I & 0 \\ -BA^{-1} & I \end{pmatrix} \begin{pmatrix} A & B^\top \\ B & C \end{pmatrix} = \begin{pmatrix} A & B^\top \\ 0 & C - BA^{-1}B^\top \end{pmatrix}.$$

The *Schur complement* of A is $D = C - BA^{-1}B^\top$; for $A = (a)$ of size 1×1, it is the Schur complement of Definition 13.9.

We use this to pivot out a whole cross at once. In round K of nested dissection, let A be the block of the current matrix indexed by the vertices of one cross, and B the block whose rows are indexed by the other vertices and whose columns are indexed by the cross.



Here A is at most $4K \times 4K$, and B has at most $8K$ nonzero rows: by Lemma 13.22, the cross vertices are adjacent only to each other and to the $8K$ vertices on the boundary of their block. So pivoting out the cross replaces the $8K \times 8K$ block of C indexed by these boundary vertices with the corresponding block of $D = C - BA^{-1}B^{\top}$. We assume a parallel dense solver that uses $O(n^{\alpha})$ work and $O(\log n)$ time on $n \times n$ matrices. For the work alone, known algorithms give $\alpha = 2.37\ldots$

Algorithm 13.24. The cost of pivoting out one cross is:

	work	time	
1) compute A^{-1}	$O(K^{\alpha})$	$O(\log K)$	
2) BA^{-1}	$O(K^{\alpha})$	$O(\log K)$	
3) $(BA^{-1})B^{\top}$	$O(K^{\alpha})$	$O(\log K)$	
	$O(K^{\alpha})$	$O(\log K)$	per cross

The work per level (round) is therefore at most

$$\Big(\frac{n}{4K^2}\Big) cK^{\alpha} \le cK^{\alpha-2}n$$

for a constant c. With $K = 2^i$ for $0 \le i \le l-1$ and $\alpha > 2$, the geometric sum gives

$$\text{Total work} \le cn \sum_{i=0}^{l-1} (2^i)^{\alpha-2} \le \frac{c}{2^{\alpha-2}-1}\, n(\sqrt{n})^{\alpha-2} = c'\, n^{\alpha/2},$$

where $c' = c/(2^{\alpha-2}-1)$. The final dense block on the $4s$ boundary vertices costs $O(s^{\alpha}) = O(n^{\alpha/2})$ more. For $\alpha = 2.37\ldots$, this is about $n^{1.19}$, compared with the $O(n^{3/2})$ of Theorem 13.21.

14 Basic Iterative Method, Extrapolation, Chebyshev Acceleration

Lecture of February 19, 2020. Based on Gary L. Miller's handwritten notes.

Iterative methods solve a linear system approximately: they improve a guess step by step, using only multiplications by the matrix. This lecture starts from the geometric series for $1/a$, which leads to Richardson's method, and analyzes its error through the eigenvalues of $I - A$. Scaling the system, the extrapolated method, makes the iteration converge for every symmetric positive definite matrix, with $O(\kappa)$ iterations per bit of accuracy, where κ is the condition number. Polynomial acceleration combines all the iterates; Chebyshev polynomials give the best combination, with $O(\sqrt{\kappa})$ iterations per bit, and a three-term recurrence turns it into an iterative algorithm. For the path P_n, using the bounds on $\lambda_2(P_n)$ from Lecture 9, the difference is $\Theta(n^2)$ against $O(n)$ iterations per bit.

14.1 The basic iterative method

Our goal is to solve $Au = b$ approximately. We start with a simpler problem: solve $a \cdot x = 1$ approximately, for a real number a, without division.

Definition 14.1. For a real number a, let $\bar{a} = 1 - a$, so that $a = 1 - \bar{a}$.

If $|\bar{a}| < 1$, or equivalently $0 < a < 2$, the geometric series gives

$$\frac{1}{a} = \frac{1}{1 - \bar{a}} = 1 + \bar{a} + \bar{a}^2 + \cdots .$$

Let

$$u^{(k)} = 1 + \bar{a} + \cdots + \bar{a}^{k-1}$$

be the partial sums, with $u^{(0)} = 0$, the empty sum. They satisfy the recurrence

$$u^{(k+1)} = 1 + \bar{a}\, u^{(k)},$$

which uses only multiplications and additions.

Suppose now that the goal is $a \cdot x = b$. Set $v^{(k)} = b\, u^{(k)}$. Multiplying the recurrence by b gives $v^{(0)} = 0$ and

$$\begin{aligned} v^{(k+1)} &= b + \bar{a}\, v^{(k)} \\ &= b + (1 - a)\, v^{(k)} = v^{(k)} + \big(b - a v^{(k)}\big). \end{aligned}$$

14.1.1 Error analysis

Proposition 14.2. *Let $a \neq 0$ and b be real numbers, and let $v^{(k)}$ be defined by $v^{(0)} = 0$ and $v^{(k+1)} = v^{(k)} + (b - a v^{(k)})$. Then the errors*

$$\varepsilon^{(k)} = \frac{b}{a} - v^{(k)}$$

satisfy $\varepsilon^{(0)} = b/a$ and $\varepsilon^{(k)} = \bar{a}^k \varepsilon^{(0)}$ for every $k \geq 0$. Hence, if $b \neq 0$, the $v^{(k)}$ converge to b/a if and only if $|\bar{a}| < 1$, that is, $0 < a < 2$.

Proof. By the computation above, $v^{(k)} = b(1 + \bar{a} + \cdots + \bar{a}^{k-1})$. If $|\bar{a}| < 1$, then

$$\begin{aligned}\varepsilon^{(k)} &= b\big(1 + \bar{a} + \bar{a}^2 + \cdots\big) - b\big(1 + \bar{a} + \cdots + \bar{a}^{k-1}\big) \\ &= b\bar{a}^k\big(1 + \bar{a} + \cdots\big) = \frac{b\bar{a}^k}{a} = \bar{a}^k\Big(\frac{b}{a}\Big) = \bar{a}^k\varepsilon^{(0)}.\end{aligned}$$

For an arbitrary $a \neq 0$, the finite geometric sum gives $v^{(k)} = b(1 - \bar{a}^k)/(1 - \bar{a}) = b(1 - \bar{a}^k)/a$, which leads to the same formula. Since $\varepsilon^{(0)} \neq 0$ when $b \neq 0$, the errors tend to 0 if and only if $|\bar{a}| < 1$. □

14.2 Richardson's method

The same recurrence makes sense for an $n \times n$ matrix A and a vector b. There are two ways to view it.

View 1. Correct the current guess by the residual:

$$u^{(m+1)} = u^{(m)} + \underbrace{\big(b - Au^{(m)}\big)}_{\text{residual error}}.$$

View 2. Iterate an affine map:

$$u^{(m+1)} = (I - A)u^{(m)} + b = Gu^{(m)} + b, \qquad G = I - A.$$

The method has two advantages:

1) each step costs $O(w(A))$ operations, where $w(A)$ is the number of nonzero entries of A, and the work parallelizes well;

2) a fixed point is a solution: $u = Gu + b$ if and only if $Au = b$.

It also has two disadvantages:

1) it may not converge;

2) it may converge very slowly.

14.3 The error for Richardson

Suppose that $A\bar{u} = b$, and let $G = I - A$.

Definition 14.3. The *error* of the iterate $u^{(m)}$ is $\varepsilon^{(m)} = u^{(m)} - \bar{u}$.

Claim 14.4. $\varepsilon^{(m+1)} = G\varepsilon^{(m)}$.

Proof. Since $A\bar{u} = b$,

$$\begin{aligned}G\varepsilon^{(m)} &= G\big(u^{(m)} - \bar{u}\big) = Gu^{(m)} - (I - A)\bar{u} \\ &= Gu^{(m)} - \bar{u} + A\bar{u} = \big(Gu^{(m)} + b\big) - \bar{u} \\ &= u^{(m+1)} - \bar{u} = \varepsilon^{(m+1)}.\end{aligned}$$

□

Hence $\varepsilon^{(m)} = G^m \varepsilon^{(0)}$. The method converges from the starting point $u^{(0)}$ if and only if

$$\lim_{m\to\infty} G^m \varepsilon^{(0)} = 0,$$

and it converges from every starting point if and only if

$$\lim_{m\to\infty} G^m = 0,$$

as we see by taking $\varepsilon^{(0)}$ to be the standard basis vectors.

If A is symmetric, then so is G, and

$$G = \sum_{i=1}^{n} \lambda_i \, v_i v_i^{\top},$$

where $\lambda_1, \dots, \lambda_n$ are the eigenvalues of G and $v_1, \dots, v_n$ are corresponding orthonormal eigenvectors. Then $G^m = \sum_i \lambda_i^m v_i v_i^{\top}$, so Richardson's method converges from every starting point if and only if $|\lambda_i| < 1$ for all i. For a general symmetric A, however, there may be an i with $|\lambda_i| \geq 1$: since the eigenvalues of G are the numbers $1 - \lambda$ for the eigenvalues λ of A, this happens whenever A has an eigenvalue outside the interval $(0, 2)$. A simple modification gives convergence for every symmetric positive definite A.

14.4 The extrapolated method

Throughout this subsection, A is symmetric positive definite, with largest eigenvalue $M = \lambda_{\max}(A)$ and smallest eigenvalue $m = \lambda_{\min}(A) > 0$. Our goal is to make Richardson's method converge.

For $\alpha \neq 0$, we have $Ax = b$ if and only if $\alpha A x = \alpha b$. Richardson's method for the scaled system is the new recurrence

$$\begin{aligned} u^{(k+1)} &= u^{(k)} + \alpha(b - Au^{(k)}), \quad \text{or} \\ u^{(k+1)} &= (I - \alpha A)u^{(k)} + \alpha b. \end{aligned}$$

Set $G_\alpha = I - \alpha A$. By Claim 14.4 applied to αA, the errors satisfy $\varepsilon^{(k)} = G_\alpha^k \varepsilon^{(0)}$. Let

$$\sigma(G) = \max_i |\lambda_i(G)|$$

be the spectral radius of a symmetric matrix G. Since $G_\alpha^k = \sum_i \lambda_i^k v_i v_i^{\top}$ in the notation above, we have $\|\varepsilon^{(k)}\| \leq \sigma(G_\alpha)^k \|\varepsilon^{(0)}\|$, with equality when $\varepsilon^{(0)}$ is an eigenvector of an eigenvalue of largest absolute value. So our goal is to choose α to achieve

$$\min_\alpha \sigma(G_\alpha).$$

Claim 14.5. *The step size* $\alpha = \dfrac{2}{M+m}$ *is optimal: it minimizes* $\sigma(G_\alpha)$ *over all real* α*, and* $\sigma(G_{2/(M+m)}) = \dfrac{M-m}{M+m}$.

The eigenvalues of G_α come from those of A: we have $Ax = \lambda_A x$ if and only if $G_\alpha x = (1 - \alpha\lambda_A)x$. More generally, the following holds.

Fact 14.6 (spectral mapping)**.** *Let G be a symmetric matrix and $Q(z)$ a polynomial. If $\lambda \in \lambda(G)$, then $Q(\lambda) \in \lambda(Q(G))$, with the same eigenvector; and every eigenvalue of $Q(G)$ arises in this way: $\lambda(Q(G)) = Q(\lambda(G))$, counted with multiplicity. The converse, that $Q(\lambda) \in \lambda(Q(G))$ implies $\lambda \in \lambda(G)$, holds when Q is one-to-one, for example when $Q(z) = 1 - \alpha z$ with $\alpha \neq 0$.*

Indeed, if $G = \sum_i \lambda_i v_i v_i^\top$, then $Q(G) = \sum_i Q(\lambda_i)\, v_i v_i^\top$.

Proof of Claim 14.5. Here $Q(z) = 1 - \alpha z$, so the eigenvalues of G_α are the numbers $1 - \alpha\lambda$ with $\lambda \in \lambda(A)$, and $m, M \in \lambda(A) \subseteq [m, M]$. For $\alpha = 2/(M+m) > 0$, the map $\lambda \mapsto 1 - \alpha\lambda$ is decreasing, so

$$\lambda_{\max}(G_\alpha) = 1 - \alpha m = 1 - \frac{2m}{M+m} = \frac{M-m}{M+m},$$
$$\lambda_{\min}(G_\alpha) = 1 - \alpha M = 1 - \frac{2M}{M+m} = \frac{m-M}{M+m} = -\frac{M-m}{M+m}.$$

Thus

$$\sigma(G_\alpha) = \frac{M-m}{M+m}.$$

For any other real α, we have $\sigma(G_\alpha) \geq \max\{|1 - \alpha m|, |1 - \alpha M|\}$. If $\alpha < 2/(M+m)$, then $1 - \alpha m > (M-m)/(M+m)$; if $\alpha > 2/(M+m)$, then $\alpha M - 1 > (M-m)/(M+m)$. In both cases $\sigma(G_\alpha) > (M-m)/(M+m)$. □

Definition 14.7. The *condition number* of a symmetric positive definite matrix A is

$$\kappa(A) \coloneqq \frac{M_A}{m_A} = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}.$$

Then, for $\alpha = 2/(M+m)$,

$$\sigma(G_\alpha) = \frac{M-m}{M+m} = \frac{\frac{M}{m} - \frac{m}{m}}{\frac{M}{m} + \frac{m}{m}} = \frac{\kappa(A) - 1}{\kappa(A) + 1}.$$

Set $\kappa = \kappa(A)$ and $\sigma = \sigma(G_\alpha)$. Since $\kappa \geq 1$,

$$\sigma = 1 - \frac{2}{\kappa + 1} \leq 1 - \frac{1}{\kappa}.$$

After κ iterations (more precisely, $\lceil \kappa \rceil$), the error has been multiplied by at most

$$\left(1 - \frac{1}{\kappa}\right)^\kappa \leq \frac{1}{e}.$$

Since $e > 2$, we need at most $\lceil \kappa \rceil$ iterations per bit of accuracy.

Remark 14.8. The bound $\sigma \leq 1 - 1/\kappa$ loses a factor of about 2: we have $\sigma = 1 - 2/(\kappa+1) \approx 1 - 2/\kappa$ for large κ, and $\sigma \leq e^{-2/(\kappa+1)}$. So after κ iterations the error has been multiplied by at most $e^{-2\kappa/(\kappa+1)} \approx e^{-2}$, and already $(\kappa+1)(\ln 2)/2$ iterations halve the error.

Example 14.9. Let $A = L(P_n)$ be the Laplacian of the path P_n with $n \geq 3$ vertices and unit weights. This matrix is singular, since $A\mathbf{1} = 0$, so we work on the vectors orthogonal to $\mathbf{1}$. If $b \perp \mathbf{1}$ and $u^{(0)} \perp \mathbf{1}$, then every iterate is orthogonal to $\mathbf{1}$, because $\mathbf{1}^\top A = 0$, and so is the error if we take the solution $\bar{u} \perp \mathbf{1}$; on this subspace the eigenvalues of A are $\lambda_2 \leq \cdots \leq \lambda_n$, so the

analysis above applies with $m = \lambda_2$, $M = \lambda_n$ and $\kappa = \lambda_n/\lambda_2$. By Lecture 9, $4/n^2 \le \lambda_2 \le 12/n^2$. Moreover $2 \le \lambda_n \le 4$: the Rayleigh quotient of the second standard basis vector is 2, and $x^\top Ax = \sum_i (x_i - x_{i+1})^2 \le 2\sum_i (x_i^2 + x_{i+1}^2) \le 4\|x\|^2$. Hence $n^2/6 \le \kappa \le n^2$. So $\kappa(A) = \Theta(n^2)$, and the extrapolated method needs $\Theta(n^2)$ iterations per bit. It does need that many: if $\varepsilon^{(0)}$ is an eigenvector of λ_2, then $\varepsilon^{(k)} = (1 - \alpha\lambda_2)^k \varepsilon^{(0)} = \sigma^k \varepsilon^{(0)}$, and $\sigma^k \ge 1 - 2k/(\kappa+1) > 1/2$ for $k < (\kappa+1)/4$, so halving this error takes at least $(\kappa+1)/4 \ge n^2/24$ iterations.

14.5 Polynomial acceleration

Assume that A is symmetric positive definite (SPD) and $A\bar{x} = b$. Richardson's method computes

$$x^{(m+1)} = Gx^{(m)} + b, \qquad G = I - A, \qquad \varepsilon^{(m)} = x^{(m)} - \bar{x} = G^m \varepsilon^{(0)}.$$

Everything below applies equally to the extrapolated method, with A and b replaced by αA and αb.

The idea is to use all previous iterates $x^{(i)}$, not only the last one:

- compute $x^{(0)}, \dots, x^{(n)}$;
- pick $\alpha_0, \dots, \alpha_n \in \mathbb{R}$ such that $\sum_{i=0}^n \alpha_i = 1$;
- return $u^{(n)} = \sum_{i=0}^n \alpha_i x^{(i)}$.

To see how to pick the α_i, we look at the error.

Definition 14.10. The error of $u^{(n)}$ is $\tilde{\varepsilon}^{(n)} = u^{(n)} - \bar{x}$, and its *error polynomial* is

$$Q_n(z) = \alpha_0 + \alpha_1 z + \cdots + \alpha_n z^n.$$

Since $\sum_i \alpha_i = 1$,

$$\begin{aligned}\tilde{\varepsilon}^{(n)} &= \sum_{i=0}^n \alpha_i x^{(i)} - \bar{x} = \sum_{i=0}^n \alpha_i (x^{(i)} - \bar{x}) \\ &= \sum_{i=0}^n \alpha_i \varepsilon^{(i)} = \sum_{i=0}^n \alpha_i G^i \varepsilon^{(0)} = Q_n(G)\, \varepsilon^{(0)}.\end{aligned}$$

The condition $\sum_i \alpha_i = 1$ says $Q_n(1) = 1$. So the possible error polynomials are exactly the polynomials of degree at most n with value 1 at 1.

Our goal is, for each n, to pick such a $Q_n(z)$ achieving

$$\min \sigma(Q_n(G)).$$

By Fact 14.6,

$$\lambda(Q_n(G)) = Q_n(\lambda(G)),$$

so $\sigma(Q_n(G)) = \max_{\lambda \in \lambda(G)} |Q_n(\lambda)|$, and $\|\tilde{\varepsilon}^{(n)}\| \le \sigma(Q_n(G))\|\varepsilon^{(0)}\|$ because $Q_n(G)$ is symmetric.

14.6 Chebyshev polynomials and acceleration

Definition 14.11 (Chebyshev polynomials)**.** The *Chebyshev polynomials* $T_0, T_1, T_2, \dots$ are defined by

$$T_0(w) = 1, \qquad T_1(w) = w,$$
$$T_{n+1}(w) = 2w\,T_n(w) - T_{n-1}(w) \qquad (n \geq 1).$$

Example 14.12.

$$T_2(w) = 2w \cdot w - 1 = 2w^2 - 1,$$
$$T_3(w) = 2w(2w^2 - 1) - w = 4w^3 - 2w - w = 4w^3 - 3w.$$

By induction, T_n has degree n.

Claim 14.13. *For every $n \geq 0$, $T_n(1) = 1$ and $T_n(-1) = (-1)^n$. In particular, the coefficients of T_n sum to 1.*

Proof. Both hold for $n = 0, 1$. If they hold for $n-1$ and n, then $T_{n+1}(1) = 2 - 1 = 1$ and $T_{n+1}(-1) = -2(-1)^n - (-1)^{n-1} = (-1)^{n+1}$. □

Theorem 14.14. *For every real number w and every $n \geq 0$,*

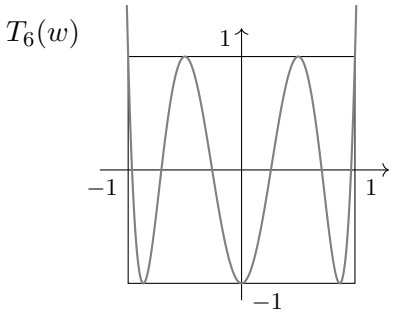
$$T_n(w) = \frac{1}{2}\left[\left(w + \sqrt{w^2 - 1}\right)^n + \left(w + \sqrt{w^2 - 1}\right)^{-n}\right],$$

where for $|w| < 1$ the square root denotes $i\sqrt{1 - w^2}$.

The number $w + \sqrt{w^2 - 1}$ is never 0, since its product with $w - \sqrt{w^2 - 1}$ is 1.

Homework 14.15. Prove Theorem 14.14.

The graph of T_6 on $[-1, 1]$ oscillates between -1 and 1, and T_6 grows quickly outside this interval:



Theorem 14.16. *Let $d > 1$ and $n \geq 1$, and let $H_n(w) = T_n(w)/T_n(d)$. Then*

1) $\max_{-1 \leq w \leq 1} |H_n(w)| = \dfrac{1}{T_n(d)}$;

2) *H_n is the minimum such polynomial: every polynomial p of degree at most n with $p(d) = 1$ satisfies $\max_{-1 \leq w \leq 1} |p(w)| \geq 1/T_n(d)$.*

Proof. By Theorem 14.14, $T_n(d) = (y^n + y^{-n})/2 \geq 1$ with $y = d + \sqrt{d^2 - 1} > 1$. For $w = \cos\theta$ we have $w + \sqrt{w^2 - 1} = \cos\theta + i\sin\theta = e^{i\theta}$, so Theorem 14.14 gives $T_n(\cos\theta) = \cos(n\theta)$. Hence $|T_n(w)| \leq 1$ on $[-1, 1]$, and at the $n + 1$ points $w_k = \cos(k\pi/n)$, $k = 0, \ldots, n$, we have $T_n(w_k) = (-1)^k$. This proves part 1.

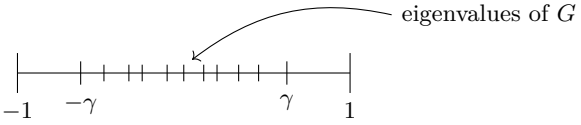
For part 2, suppose that p has degree at most n, $p(d) = 1$, and $|p(w)| < 1/T_n(d)$ on $[-1, 1]$. Then $R = H_n - p$ has degree at most n, and $R(w_k)$ has the sign of $(-1)^k$, because $|H_n(w_k)| = 1/T_n(d) > |p(w_k)|$. So R has a root in each of the n intervals (w_{k+1}, w_k), and also the root $d > 1$. A nonzero polynomial of degree at most n has at most n roots, so $R = 0$, which contradicts $R(w_0) > 0$. □

14.7 Chebyshev plus extrapolated method

Let A be SPD with condition number $\kappa = \kappa(A) > 1$, and let

$$\gamma = \frac{\kappa - 1}{\kappa + 1} = \frac{M - m}{M + m} \in (0, 1).$$

We apply polynomial acceleration to the extrapolated method with $\alpha = 2/(M + m)$, so from now on $G = G_\alpha = I - \alpha A$. By the proof of Claim 14.5, the eigenvalues of G lie in $[-\gamma, \gamma]$, and $\pm\gamma$ are eigenvalues:



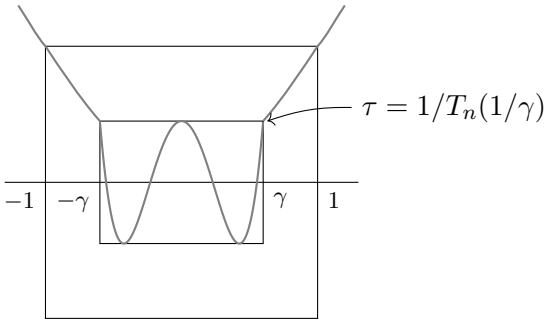
For our polynomial acceleration we use

$$\tilde{T}_n(x) = \frac{T_n(x/\gamma)}{T_n(1/\gamma)},$$

which is well defined because $T_n(1/\gamma) \geq 1$ by the proof of Theorem 14.16. Since $\tilde{T}_n(1) = 1$, the coefficients of $\tilde{T}_n$ sum to 1, so $\tilde{T}_n$ is a valid error polynomial of degree n. Let

$$\tau = \frac{1}{T_n(1/\gamma)}.$$

On $[-\gamma, \gamma]$, the graph of $\tilde{T}_n$ stays between $-\tau$ and τ:



(The picture is not to scale: τ is tiny for large n.) Since $|\lambda(G)| \le \gamma$, part 1 of Theorem 14.16 gives $|\lambda(\tilde{T}_n(G))| \le \tau$, that is, $\sigma(\tilde{T}_n(G)) \le \tau$. In fact equality holds, since γ is an eigenvalue of G and $\tilde{T}_n(\gamma) = \tau$.

Our goal is now to bound $T_n(1/\gamma)$ from below. We have

$$\frac{1}{\gamma} = \frac{\kappa+1}{\kappa-1} = 1 + \frac{2}{\kappa-1} = 1 + 2\mu, \qquad \text{where } \mu = \frac{1}{\kappa-1}.$$

For $x \ge 1$, Theorem 14.14 gives

$$\begin{aligned} T_n(x) &= \frac{1}{2}\Big[(x+\sqrt{x^2-1})^n + (x+\sqrt{x^2-1})^{-n}\Big] \\ &\ge \frac{1}{2}(x+\sqrt{x^2-1})^n, \end{aligned}$$

since the second term is positive. So

$$\begin{aligned} T_n(1+2\mu) &\ge \frac{1}{2}\Big(1+2\mu+\sqrt{(1+2\mu)^2-1}\Big)^n \\ &= \frac{1}{2}\Big(1+2\mu+2\sqrt{\mu(\mu+1)}\Big)^n \\ &= \frac{1}{2}(\sqrt{\mu}+\sqrt{\mu+1})^{2n}. \end{aligned} \tag{$*$}$$

Since $\mu = 1/(\kappa-1)$ and $\mu+1 = \kappa/(\kappa-1)$,

$$(\sqrt{\mu}+\sqrt{\mu+1})^2 = \Big(\frac{1}{\sqrt{\kappa-1}} + \frac{\sqrt{\kappa}}{\sqrt{\kappa-1}}\Big)^2 = \frac{(1+\sqrt{\kappa})^2}{\kappa-1} = \frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}.$$

Thus the right-hand side of $(*)$ equals $\frac{1}{2}(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1})^n$, and finally

$$T_n(1/\gamma) \ge \frac{1}{2}\Big(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\Big)^n, \qquad \text{or} \qquad \tau \le 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^n.$$

Theorem 14.17. *Let A be SPD with $\kappa = \kappa(A) > 1$, and let*

$$\rho = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} = 1 - \frac{2}{\sqrt{\kappa}+1}.$$

The extrapolated method with Chebyshev acceleration ($\alpha = 2/(M+m)$ and $Q_n = \tilde{T}_n$) satisfies $\|\tilde{\varepsilon}^{(n)}\| \le \sigma(\tilde{T}_n(G))\,\|\varepsilon^{(0)}\|$ and

$$\rho^n \le \sigma(\tilde{T}_n(G)) \le 2\rho^n.$$

In this sense, the convergence rate $1-\rho$ of extrapolated Chebyshev is $\Theta(1/\sqrt{\kappa})$.

Proof. The first inequality and the upper bound were shown above. For the lower bound, $\sigma(\tilde{T}_n(G)) = \tau$, and for $x \ge 1$ Theorem 14.14 gives $T_n(x) \le (x+\sqrt{x^2-1})^n$; with $x = 1/\gamma$ this is $(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1})^n$ by the computation above, so $\tau \ge \rho^n$. □

Since $2\rho^n \le 2e^{-2n/(\sqrt{\kappa}+1)}$, Chebyshev acceleration needs only $O(\sqrt{\kappa})$ iterations per bit, against $O(\kappa)$ for the extrapolated method alone.

Back to P_n: by Example 14.9, $\kappa(L(P_n)) = \Theta(n^2)$. Therefore the convergence rate of Chebyshev acceleration is $\Theta(1/n)$: $O(n)$ iterations per bit.

Remark 14.18. Up to constant factors, this is optimal for a multi-term recurrence, since such a method performs only local operations: each iteration multiplies by A once, and on P_n this moves information along one edge. So after k iterations from $u^{(0)} = 0$, the iterate at a vertex depends only on the entries of b within distance k of that vertex, and $\Omega(n)$ iterations are needed before the iterate at one end of the path can depend on the entries of b at the other end.

14.8 From polynomial recurrence to iterative algorithm

Computing $u^{(n)} = \sum_i \alpha_i x^{(i)}$ directly would require storing all iterates. A three-term recurrence for the polynomials avoids this. Consider polynomials defined by

$$\begin{aligned} Q_0(x) &= 1, \\ Q_1(x) &= \alpha_0 x + \beta_0, && \alpha_0 + \beta_0 = 1, \\ Q_{n+1}(x) &= \alpha_n x\, Q_n(x) + \beta_n Q_{n-1}(x), && \alpha_n + \beta_n = 1 \qquad (n \geq 1). \end{aligned}$$

By induction, each Q_n has degree at most n and $Q_n(1) = 1$, so it is a valid error polynomial.

Algorithm 14.19 (iterative algorithm)**.** Given $u^{(0)}$, set

$$u^{(1)} = \alpha_0 (G u^{(0)} + b) + \beta_0 u^{(0)},$$

and for $n \geq 1$,

$$u^{(n+1)} = \alpha_n (G u^{(n)} + b) + \beta_n u^{(n-1)}.$$

(For the extrapolated method, $G = G_\alpha$ and b is replaced by αb.)

Each step costs one multiplication by G, that is, $O(w(A))$ operations, and uses only the two previous iterates.

Proposition 14.20. *Let $A\bar{u} = b$ and $\varepsilon^{(n)} = u^{(n)} - \bar{u}$ for the iterates of Algorithm 14.19. Then $\varepsilon^{(n)} = Q_n(G)\,\varepsilon^{(0)}$ for every $n \geq 0$.*

Proof. We induct on n, using $G\bar{u} = \bar{u} - A\bar{u} = \bar{u} - b$ and $\alpha_n + \beta_n = 1$. The case $n = 0$ is trivial. For $n = 1$,

$$\varepsilon^{(1)} = \alpha_0 (G u^{(0)} - G\bar{u}) + \beta_0 (u^{(0)} - \bar{u}) = (\alpha_0 G + \beta_0 I)\varepsilon^{(0)} = Q_1(G)\,\varepsilon^{(0)}.$$

Suppose the claim holds for $n-1$ and n. Then

$$\begin{aligned} Q_{n+1}(G)\,\varepsilon^{(0)} &= \left[\alpha_n G\, Q_n(G) + \beta_n Q_{n-1}(G)\right]\varepsilon^{(0)} \\ &= \alpha_n G(\varepsilon^{(n)}) + \beta_n (\varepsilon^{(n-1)}) \\ &= \alpha_n G(u^{(n)} - \bar{u}) + \beta_n (u^{(n-1)} - \bar{u}) \\ &= \alpha_n (G u^{(n)} - \bar{u} + b) + \beta_n u^{(n-1)} - \beta_n \bar{u} \\ &= \alpha_n (G u^{(n)} + b) + \beta_n u^{(n-1)} - \bar{u} \\ &= u^{(n+1)} - \bar{u} = \varepsilon^{(n+1)}. \end{aligned}$$

□

Remark 14.21. The polynomials $\tilde{T}_n$ of Section 14.7 fit this scheme. By the recurrence of Definition 14.11 at x/γ, $\tilde{T}_{n+1}(x) = \alpha_n x\, \tilde{T}_n(x) + \beta_n \tilde{T}_{n-1}(x)$ for $n \geq 1$, with

$$\alpha_n = \frac{2\,T_n(1/\gamma)}{\gamma\, T_{n+1}(1/\gamma)}, \qquad \beta_n = -\frac{T_{n-1}(1/\gamma)}{T_{n+1}(1/\gamma)},$$

and $\alpha_n + \beta_n = 1$ by the same recurrence at $1/\gamma$; also $\tilde{T}_1(x) = x$, so $\alpha_0 = 1$ and $\beta_0 = 0$. So the Chebyshev-accelerated method runs as Algorithm 14.19, with $O(w(A))$ work per step.

15 Conjugate Gradient Method and Steepest Descent

Lecture of February 21, 2020. Based on Gary L. Miller's handwritten notes.

This lecture begins the study of gradient methods for solving $Ax = b$ with A symmetric positive definite, which continues with the conjugate gradient method in Lecture 16. Solving the system is equivalent to minimizing a convex quadratic function F whose gradient is $Ax - b$, so Richardson's method from Lecture 14 is gradient descent with a fixed step size. Steepest descent instead chooses each step size by minimizing F along the search direction. Using the Kantorovich inequality, we show that each step of steepest descent reduces the error in the A-norm by the factor $(\kappa - 1)/(\kappa + 1)$, where κ is the condition number. A two-dimensional example shows a single step in detail.

Throughout this lecture, A is an $n \times n$ symmetric positive definite (SPD) matrix, $b \in \mathbb{R}^n$, and $\bar{x} = A^{-1}b$ is the solution of $Ax = b$.

15.1 Linear systems as minimization problems

We start with steepest descent. Consider the *elliptical bowl*

$$M(x) = (\bar{x} - x)^\top A(\bar{x} - x), \qquad x \in \mathbb{R}^n.$$

Since A is positive definite, $M(x) \geq 0$, with equality only at $x = \bar{x}$. So $\bar{x}$ is the unique minimizer of M. The level sets of M are ellipsoids centered at $\bar{x}$, and their principal axes are the eigenvectors of A.

Expanding, and using $A\bar{x} = b$ and the symmetry of A,

$$\begin{aligned} M(x) &= \bar{x}^\top A\bar{x} - 2x^\top A\bar{x} + x^\top Ax \\ &= \underbrace{\bar{x}^\top b}_{\text{constant}} - 2x^\top b + x^\top Ax. \end{aligned}$$

Hence $M(x) = 2F(x) + \bar{x}^\top b$, where

$$F(x) = \frac{1}{2}\, x^\top Ax - x^\top b,$$

and it suffices to minimize F: its unique minimizer is $\bar{x}$.

15.2 Crash course on matrix calculus

Let $G\colon \mathbb{R}^n \to \mathbb{R}$ be a differentiable function, $G(x_1, \ldots, x_n) = y$.

Definition 15.1. The *gradient* of G is $\nabla G = \left(\dfrac{\partial G}{\partial x_1}, \ldots, \dfrac{\partial G}{\partial x_n}\right)$.

The gradient is linear, so

$$\nabla F(u) = \frac{1}{2}\nabla(u^\top Au) - \nabla(u^\top b).$$

Claim 15.2. $\dfrac{\partial\, u^\top b}{\partial u_i} = b_i$.

Proof. With e_i the ith standard basis vector,

$$\frac{\partial\, u^\top b}{\partial u_i} = \lim_{h\to 0} \frac{(u+he_i)^\top b - u^\top b}{h} = \lim_{h\to 0} \frac{he_i^\top b}{h} = b_i. \qquad \square$$

Similarly, since A is symmetric,

$$\frac{\partial\, u^\top Au}{\partial u_i} = 2(Au)_i.$$

Indeed, $(u+he_i)^\top A(u+he_i) - u^\top Au = 2h\, e_i^\top Au + h^2 a_{ii}$, using $u^\top Ae_i = e_i^\top Au$; dividing by h and letting $h \to 0$ gives $2(Au)_i$. Thus

$$\nabla F(u) = Au - b.$$

In particular, $-\nabla F(u) = b - Au$ is the residual of u, and $\nabla F(u) = 0$ exactly when $u = \bar{x}$.

15.3 Richardson's algorithm and a variable step size

Richardson's algorithm is

$$x^{(n+1)} = x^{(n)} + \alpha\big(b - Ax^{(n)}\big), \qquad \alpha = 1.$$

Since $b - Ax^{(n)} = -\nabla F(x^{(n)})$, each step moves in the direction of steepest descent of F. Thus Richardson's algorithm is steepest descent with the fixed step size $\alpha = 1$, and the extrapolated method of Lecture 14 is the optimal fixed-size method. Our goal is a variable-size method.

The idea is to pick, at each step, the $\alpha \in \mathbb{R}$ that minimizes $F\big(x^{(n)} + \alpha g\big)$, where $g = \nabla F(x^{(n)})$. We do this for a general search direction. Let

1) u be some guess,

2) $d \neq 0$ a search direction,

3) α the step size, and

4) $g = \nabla F(u) = Au - b$ the gradient at u.

Then

$$\begin{aligned} F(u+\alpha d) &= \frac{1}{2}(u+\alpha d)^\top A(u+\alpha d) - b^\top(u+\alpha d) \\ &= \frac{1}{2}u^\top Au + \alpha u^\top Ad + \frac{1}{2}\alpha^2 d^\top Ad - b^\top u - \alpha b^\top d, \end{aligned}$$

so

$$\begin{aligned} \frac{dF(u+\alpha d)}{d\alpha} &= u^\top Ad + \alpha d^\top Ad - b^\top d \\ &= d^\top(Au - b) + \alpha d^\top Ad \\ &= d^\top g + \alpha d^\top Ad. \end{aligned}$$

As a function of α, $F(u+\alpha d)$ is a quadratic polynomial with leading coefficient $\frac{1}{2}d^\top Ad > 0$, so it is minimized exactly where its derivative vanishes:

$$d^\top g + \alpha d^\top Ad = 0, \qquad \text{that is,} \qquad \alpha = \frac{-d^\top g}{d^\top Ad}.$$

Claim 15.3. *For this α, the new gradient is orthogonal to the search direction: $\nabla F(u+\alpha d) \perp d$.*

Proof.

$$\begin{aligned} d^\top (A(u+\alpha d) - b) &= d^\top (Au - b) + \alpha d^\top A d \\ &= d^\top g - d^\top g = 0. \end{aligned}$$

□

15.4 Steepest descent

Steepest descent applies this line search with $d = g$.

Algorithm 15.4 (steepest descent)**.**

Input: A, b, with A SPD.

Init: $x^{(0)} = 0$.

Iteration: for $n = 0, 1, 2, \ldots$, let $g = Ax^{(n)} - b$; if $g = 0$, stop. Otherwise set

$$x^{(n+1)} = x^{(n)} + \alpha g, \qquad \text{where} \quad \alpha = -\, g^\top g / g^\top A g.$$

Since $\alpha < 0$, each step moves along $-g$, the direction of steepest descent, and when the algorithm stops, $x^{(n)} = \bar{x}$.

15.5 Convergence rate

Lemma 15.5 (Kantorovich lemma)**.** *Let B be SPD, with largest and smallest eigenvalues λ_M and λ_m. Then for every $x \neq 0$,*

$$\frac{(x^\top B x)(x^\top B^{-1} x)}{(x^\top x)^2} \leq \frac{(\lambda_M + \lambda_m)^2}{4 \lambda_M \lambda_m}.$$

We do not prove the lemma here; see [Saa96, Chapter 5, p. 132].

Definition 15.6. The *A-norm* of $x \in \mathbb{R}^n$ is $\|x\|_A = (x^\top A x)^{1/2}$, and x, y are *A-orthogonal*, written $x \perp_A y$, if $x^\top A y = 0$.

Let $A^{1/2}$ be the SPD square root of A: if $A = U \Lambda U^\top$ with U orthogonal and Λ diagonal, then $A^{1/2} = U \Lambda^{1/2} U^\top$. Then $x^\top A y = (A^{1/2} x)^\top (A^{1/2} y)$, so $\|x\|_A = \|A^{1/2} x\|$, and $x \perp_A y$ if and only if $A^{1/2} x \perp A^{1/2} y$.

Theorem 15.7. *Let λ_M and λ_m be the largest and smallest eigenvalues of A, let $\kappa = \lambda_M / \lambda_m$, and let $\varepsilon^{(k)} = \bar{x} - x^{(k)}$ be the errors of steepest descent. Then, for every k such that $x^{(k+1)}$ is defined,*

$$\|\varepsilon^{(k+1)}\|_A \leq \Big(\frac{\lambda_M - \lambda_m}{\lambda_M + \lambda_m}\Big) \|\varepsilon^{(k)}\|_A = \Big(\frac{\kappa - 1}{\kappa + 1}\Big) \|\varepsilon^{(k)}\|_A.$$

Proof. See also [Saa96, Chapter 5, p. 133]. Write $\varepsilon = \varepsilon^{(k)}$ and let $r = -g = b - Ax^{(k)}$, which is nonzero because $x^{(k+1)}$ is defined. Then $r = A\varepsilon$, and the step is $x^{(k+1)} = x^{(k)} + tr$ with $t = r^{\top} r / r^{\top} A r$, so $\varepsilon^{(k+1)} = \varepsilon - tr$. Hence

$$\begin{aligned} \|\varepsilon^{(k+1)}\|_A^2 &= \varepsilon^{\top} A \varepsilon - 2t\, r^{\top} A \varepsilon + t^2\, r^{\top} A r = \|\varepsilon\|_A^2 - 2t\, r^{\top} r + t^2\, r^{\top} A r \\ &= \|\varepsilon\|_A^2 - \frac{(r^{\top} r)^2}{r^{\top} A r} = \|\varepsilon\|_A^2 \Big(1 - \frac{(r^{\top} r)^2}{(r^{\top} A r)(r^{\top} A^{-1} r)}\Big), \end{aligned}$$

where the last step uses $\|\varepsilon\|_A^2 = \varepsilon^{\top} A \varepsilon = r^{\top} A^{-1} r$. By Lemma 15.5 with $B = A$ and $x = r$,

$$\|\varepsilon^{(k+1)}\|_A^2 \le \|\varepsilon\|_A^2 \Big(1 - \frac{4\lambda_M \lambda_m}{(\lambda_M + \lambda_m)^2}\Big) = \Big(\frac{\lambda_M - \lambda_m}{\lambda_M + \lambda_m}\Big)^2 \|\varepsilon\|_A^2. \qquad \square$$

Remark 15.8. The factor is $(\kappa - 1)/(\kappa + 1) = 1 - 2/(\kappa + 1)$, which is about $1 - 2/\kappa$ for large κ: the same factor as the optimal fixed step of the extrapolated method in Lecture 14.

15.6 Steepest descent: an example

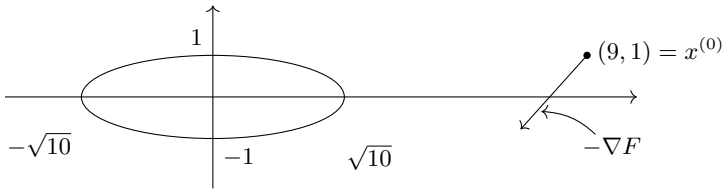
Example 15.9. Let $n = 2$ and

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \bar{x} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad x^{(0)} = \begin{pmatrix} 9 \\ 1 \end{pmatrix}.$$

The residual is

$$r = b - Ax^{(0)} = -\begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix} \begin{pmatrix} 9 \\ 1 \end{pmatrix} = -\begin{pmatrix} 9 \\ 10 \end{pmatrix},$$

so the gradient is $g = -r = (9, 10)^{\top}$. The level curves of the elliptical bowl are the ellipses $x_1^2 + 10 x_2^2 = c$; the figure shows the one with $c = 10$, and the direction $-\nabla F$ of the first step:



The step size of Algorithm 15.4 is

$$\alpha = -\frac{g^{\top} g}{g^{\top} A g} = -\frac{181}{81 + 1000} = -\frac{181}{1081},$$

so

$$x^{(1)} = x^{(0)} + \alpha g = \begin{pmatrix} 9 \\ 1 \end{pmatrix} - \Big(\frac{181}{1081}\Big) \begin{pmatrix} 9 \\ 10 \end{pmatrix}.$$

Its second coordinate is

$$x_2^{(1)} = 1 - \frac{1810}{1081} = \frac{-729}{1081} \approx -0.67.$$

We overshot: the second coordinate of the minimizer $\bar{x}$ is 0, and the first step carries the second coordinate from 1 past 0.

16 Conjugate Gradient Continued

Lecture of February 24, 2020. Based on Gary L. Miller's handwritten notes.

This lecture develops the conjugate gradient (CG) method for $Ax = b$ with A symmetric positive definite. The iterative methods of Lectures 14 and 15, started at 0, produce iterates in the Krylov subspaces $\mathcal{K}_n = \langle b, Ab, \ldots, A^{n-1}b\rangle$, and CG finds the point of $\mathcal{K}_n$ closest to the solution in the A-norm. We state the CG algorithm and its invariants (orthogonal gradients, A-orthogonal search directions), give its form with one matrix-vector product per iteration, and prove that CG minimizes the A-norm of the error over $\mathcal{K}_n$ and finds the exact solution of an $N \times N$ system in at most N iterations. Writing the error as a polynomial in A applied to the initial error and choosing a Chebyshev polynomial, we show that after n iterations CG has reduced the A-norm of the error by the factor $2(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1})^n$ of Chebyshev acceleration (Lecture 14).

Throughout this lecture, A is an $N \times N$ symmetric positive definite (SPD) matrix, $b \in \mathbb{R}^N$ is nonzero, and $\bar{x} = A^{-1}b$, so that $A\bar{x} = b$. As in Lecture 15, $F(x) = \frac{1}{2}x^\top Ax - x^\top b$, its gradient is $\nabla F(x) = Ax - b$, and $\|y\|_A = \sqrt{y^\top Ay}$ is the A-norm.

16.1 Krylov subspaces

Let $x^{(0)}$ be an initial guess for the solution of $Ax = b$.

Definition 16.1. For $n \geq 1$, the *nth Krylov subspace* is

$$\mathcal{K}_n = \langle b, Ab, A^2b, \ldots, A^{n-1}b\rangle,$$

the span of the vectors listed, and $\mathcal{K}_0 = \{0\}$. If $x^{(0)} \neq 0$, we consider instead the affine space $x^{(0)} + \mathcal{K}_n(A, r_0)$, the *Krylov space*, where $r_0 = b - Ax^{(0)}$ and $\mathcal{K}_n(A, r_0) = \langle r_0, Ar_0, \ldots, A^{n-1}r_0\rangle$ is the Krylov subspace of r_0 in place of b.

We assume that $x^{(0)} = 0$. The spaces are nested: $\mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \cdots$. The following claim explains why the spaces $\mathcal{K}_n$ are of interest.

Claim 16.2. *Consider an iterative method of the form*

$$x^{(n+1)} = \alpha_n x^{(n)} + \beta_n(b - Ax^{(n)}),$$

with real numbers α_n, β_n and $x^{(0)} = 0$. Then $x^{(n)} \in \mathcal{K}_n$ for every $n \geq 0$.

Proof. Induction on n. For $n = 0$, $x^{(0)} = 0 \in \mathcal{K}_0$.

Claim 16.3. *If $x \in \mathcal{K}_n$, then $Ax \in \mathcal{K}_{n+1}$.*

Indeed, if

$$x = a_0 b + a_1 Ab + \cdots + a_{n-1}A^{n-1}b, \quad \text{then} \quad Ax = a_0 Ab + a_1 A^2 b + \cdots + a_{n-1}A^n b,$$

so $Ax \in \mathcal{K}_{n+1}$. Now if $x^{(n)} \in \mathcal{K}_n$, then $x^{(n)}$, b and $Ax^{(n)}$ all lie in $\mathcal{K}_{n+1}$, and hence so does $x^{(n+1)}$. □

Thus all these methods do is find a point $x^{(n)} \in \mathcal{K}_n$. The best would be to find

$$x^{(n)} = \arg\min_{x \in \mathcal{K}_n} \|\bar{x} - x\|_2,$$

but we do not know how to do this efficiently, since $\bar{x}$ is unknown. The next best thing is

$$x^{(n)} = \arg\min_{x \in \mathcal{K}_n} \|\bar{x} - x\|_A.$$

This we can do, and the method is called *conjugate gradient*. The reason is that $\|\bar{x} - x\|_A^2 = 2F(x) + \bar{x}^\top b$ by the expansion in Lecture 15, so minimizing it does not require knowing $\bar{x}$.

16.2 Conjugate gradient

CG has two design goals:

1) the step directions should be A-orthogonal;

2) each step size should minimize F along the step direction, as in Lecture 15.

Algorithm 16.4 (CG($Ax = b$))**.**

Init: $x_0 = 0$, $g_0 = Ax_0 - b = -b$, $d_0 = -g_0$, and

$$x_1 = x_0 + \alpha_0 d_0, \qquad \text{where} \quad \alpha_0 = \frac{g_0^\top g_0}{g_0^\top A g_0}.$$

Iteration: for $k = 1, 2, \ldots$, compute $g_k = Ax_k - b$; if $g_k = 0$, stop and return x_k. Otherwise compute

$$\begin{aligned}
\beta_k &= \frac{g_k^\top A d_{k-1}}{d_{k-1}^\top A d_{k-1}} && \text{(correction)} \\
d_k &= -g_k + \beta_k d_{k-1} && \text{(direction)} \\
\alpha_k &= -\frac{g_k^\top d_k}{d_k^\top A d_k} && \text{(step size)} \\
x_{k+1} &= x_k + \alpha_k d_k && \text{(new guess)}
\end{aligned}$$

Here g_k is the gradient of F at x_k. The choice of β_k makes $d_k^\top A d_{k-1} = -g_k^\top A d_{k-1} + \beta_k d_{k-1}^\top A d_{k-1} = 0$, and α_k is the step size of Lecture 15 that minimizes $F(x_k + \alpha d_k)$; for $k = 0$ it reduces to α_0, since $d_0 = -g_0$. The first step is defined because $g_0 = -b \neq 0$.

Theorem 16.5. *Run CG on $Ax = b$, with A SPD and $b \neq 0$. Let $k \geq 1$, and suppose that $g_0, \ldots, g_{k-1}$ are nonzero, so that $x_1, \ldots, x_k$ and $d_0, \ldots, d_{k-1}$ are defined. Then*

$$\begin{aligned}
\mathcal{K}_k = \langle x_1, \ldots, x_k \rangle &= \langle d_0, \ldots, d_{k-1} \rangle, \\
\langle g_0, \ldots, g_{k-1} \rangle &= \langle b, \ldots, A^{k-1} b \rangle, \\
g_k^\top g_j &= 0 \quad \textit{for all } j < k,
\end{aligned}$$

and, if $g_k \neq 0$, also $d_k^\top A d_j = 0$ for all $j < k$.

The proof is much easier using the one-matrix-vector form of CG below. It is an induction on k, which we do not give here; see [TB97, Chapter 38].

Remark 16.6. We use three consequences of Theorem 16.5. First, $g_k \perp \mathcal{K}_k$, because g_k is orthogonal to $g_0, \dots, g_{k-1}$, which span $\mathcal{K}_k$. Second, $\dim \mathcal{K}_k = k$, because $g_0, \dots, g_{k-1}$ are nonzero and pairwise orthogonal. Third, if $g_k \neq 0$, then $d_k \neq 0$: since $d_{k-1} \in \mathcal{K}_k$, we have $g_k^\top d_{k-1} = 0$, so $\|d_k\|^2 = \|g_k\|^2 + \beta_k^2 \|d_{k-1}\|^2 > 0$. In particular, the divisions in CG are well defined.

16.3 The one-matrix-vector form of CG

Algorithm 16.7 (1-matrix-vector CG)**.**

Init: $x_0 = 0$, $g_0 = -b$, and $d_0 = -g_0$.

Iteration: for $n = 1, 2, \dots,$ while $g_{n-1} \neq 0$:

1) $\alpha_n = g_{n-1}^\top g_{n-1} / d_{n-1}^\top A d_{n-1}$ (step size)
2) $x_n = x_{n-1} + \alpha_n d_{n-1}$ (approximate solution)
3) $g_n = g_{n-1} + \alpha_n A d_{n-1}$ (gradient)
4) $\beta_n = g_n^\top g_n / g_{n-1}^\top g_{n-1}$ (improvement)
5) $d_n = -g_n + \beta_n d_{n-1}$ (search direction)

The work per iteration is

- one matrix-vector product, Ad_{n-1}, used in steps 1) and 3);
- two inner products, $g_n^\top g_n$ and $d_{n-1}^\top A d_{n-1}$;

plus a few vector additions and scalings. We prove one property of the iteration.

Claim 16.8. *In Algorithm 16.7, $g_n = Ax_n - b$ for every n.*

Proof. Induct on n. For $n = 0$: $g_0 = -b$ and $x_0 = 0$, so $g_0 = Ax_0 - b$. Assume the claim for $n - 1$. Then

$$\begin{aligned} Ax_n - b &= A(x_{n-1} + \alpha_n d_{n-1}) - b \\ &= Ax_{n-1} - b + \alpha_n A d_{n-1} \\ &= g_{n-1} + \alpha_n A d_{n-1} \\ &= g_n. \end{aligned}$$

□

Remark 16.9. Algorithm 16.7 computes the same iterates as Algorithm 16.4: x_n, g_n and d_n agree, and the step size α_n of Algorithm 16.7 is the step size α_{n-1} of Algorithm 16.4. Suppose that both have produced the same vectors up to index $n - 1$. By Remark 16.6, $g_{n-1}^\top d_{n-2} = 0$ if $n \geq 2$ (and $d_0 = -g_0$ if $n = 1$), so $-g_{n-1}^\top d_{n-1} = g_{n-1}^\top g_{n-1}$, and the CG step size for d_{n-1} equals the α_n of step 1). Then x_n agrees, and so does $g_n = Ax_n - b$, by Claim 16.8. By step 3), $Ad_{n-1} = (g_n - g_{n-1})/\alpha_n$, and $g_n^\top g_{n-1} = 0$ by Theorem 16.5; hence $g_n^\top A d_{n-1} = g_n^\top g_n / \alpha_n$ and $d_{n-1}^\top A d_{n-1} = g_{n-1}^\top g_{n-1} / \alpha_n$, so the correction β_n of CG equals the β_n of step 4), and d_n agrees.

Theorem 16.10. *Run CG on $Ax = b$, where A is an $N \times N$ SPD matrix and $b \neq 0$, and let $\varepsilon_n = \bar{x} - x_n$.*

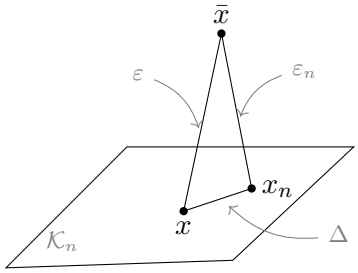
1) If $g_0, \dots, g_{n-1}$ are nonzero ($n \geq 1$), then x_n is the unique point of $\mathcal{K}_n$ that minimizes $\|\bar{x} - x\|_A$ over $x \in \mathcal{K}_n$.

2) $\|\varepsilon_n\|_A \leq \|\varepsilon_{n-1}\|_A$ whenever x_n is defined.

3) There exists $n \leq N$ such that $\varepsilon_n = 0$.

Proof of 1). By Theorem 16.5, $x_n \in \mathcal{K}_n$. Let $x \in \mathcal{K}_n$ be arbitrary, and set

$$\Delta = x_n - x \qquad \text{and} \qquad \varepsilon = \bar{x} - x = \varepsilon_n + \Delta.$$



Then

$$\|\varepsilon\|_A^2 = (\varepsilon_n + \Delta)^\top A(\varepsilon_n + \Delta) = \varepsilon_n^\top A\varepsilon_n + \Delta^\top A\Delta + 2\varepsilon_n^\top A\Delta.$$

Now $A\varepsilon_n = A(\bar{x} - x_n) = b - Ax_n = -g_n$, so $2\varepsilon_n^\top A\Delta = -2g_n^\top \Delta = 0$, since $\Delta \in \mathcal{K}_n$ and $g_n \perp \mathcal{K}_n$ (Remark 16.6). Thus

$$\|\varepsilon\|_A^2 = \|\varepsilon_n\|_A^2 + \Delta^\top A\Delta.$$

Since A is positive definite, $\Delta^\top A\Delta > 0$ whenever $x \neq x_n$, and then $\|\varepsilon\|_A^2 > \|\varepsilon_n\|_A^2$. □

Proof of 2). Since $x_{n-1} \in \mathcal{K}_{n-1} \subseteq \mathcal{K}_n$, part 1) gives $\|\varepsilon_n\|_A \leq \|\bar{x} - x_{n-1}\|_A = \|\varepsilon_{n-1}\|_A$. □

Proof of 3). By the Cayley–Hamilton theorem, there exists $n \leq N$ such that $\bar{x} \in \mathcal{K}_n$. Indeed, let $p(z) = z^N + c_{N-1}z^{N-1} + \cdots + c_0$ be the characteristic polynomial of A. Then $p(A) = 0$ and $c_0 = (-1)^N \det A \neq 0$, so $A^{-1} = -c_0^{-1}(A^{N-1} + c_{N-1}A^{N-2} + \cdots + c_1 I)$, and $\bar{x} = A^{-1}b \in \mathcal{K}_N$. If CG stops at some step $k < N$, then $g_k = 0$ and $\varepsilon_k = 0$. Otherwise x_N is defined, and by part 1) it minimizes $\|\bar{x} - x\|_A$ over $\mathcal{K}_N$, which contains $\bar{x}$; so $\varepsilon_N = 0$. □

We now turn to polynomials.

Definition 16.11. For $n \geq 1$, let $\mathcal{P}_n = \{P(z) \mid P(0) = 1 \text{ and } \deg(P) \leq n - 1\}$.

16.4 CG and polynomial approximation

The problem is to find a polynomial P_n such that $P_n(0) = 1$ and $\deg(P_n) \leq n$, to minimize $\|P_n(A)\|_A$, the operator norm of $P_n(A)$ with respect to the A-norm. The next theorem shows that CG solves this problem for the particular initial error $\varepsilon_0 = \bar{x} - x_0 = \bar{x}$, which is nonzero.

Theorem 16.12. *Run CG, and suppose that $g_0, \dots, g_{n-1}$ are nonzero ($n \geq 1$). Then there exists a unique $P_n \in \mathcal{P}_{n+1}$ such that*

1) $\varepsilon_n = \bar{x} - x_n = P_n(A)\,\varepsilon_0$, and

2)

$$\frac{\|\varepsilon_n\|_A}{\|\varepsilon_0\|_A} = \inf_{P\in\mathcal{P}_{n+1}} \frac{\|P(A)\,\varepsilon_0\|_A}{\|\varepsilon_0\|_A} \le \inf_{P\in\mathcal{P}_{n+1}} \max_{\lambda\in\lambda(A)} |P(\lambda)|.$$

Proof of 1). By Theorem 16.5, $x_n \in \mathcal{K}_n$, where $x_0 = 0$. Thus $x_n = Q_{n-1}(A)\,b$ for some polynomial $Q_{n-1}(z)$ with $\deg(Q_{n-1}) \le n-1$. Consider

$$P_n(z) = 1 - z\cdot Q_{n-1}(z), \qquad \deg(P_n) \le n;$$

since $P_n(0) = 1$, we have $P_n \in \mathcal{P}_{n+1}$.

Claim 16.13. *$\varepsilon_n = P_n(A)\,\varepsilon_0$, where $\varepsilon_0 = \bar{x}$.*

Indeed,

$$\begin{aligned} P_n(A)\,\varepsilon_0 &= (I - A\,Q_{n-1}(A))\bar{x} = \bar{x} - Q_{n-1}(A)(A\bar{x}) \\ &= \bar{x} - Q_{n-1}(A)\,b = \bar{x} - x_n = \varepsilon_n. \end{aligned}$$

For uniqueness, suppose that $P \in \mathcal{P}_{n+1}$ also satisfies $P(A)\,\varepsilon_0 = \varepsilon_n$. Since $P(0) = P_n(0) = 1$, we can write $P_n(z) - P(z) = zR(z)$ with $\deg(R) \le n-1$, and then $0 = R(A)\,A\bar{x} = R(A)\,b$. By Remark 16.6, the vectors $b, Ab, \ldots, A^{n-1}b$ are linearly independent, so $R = 0$ and $P = P_n$. □

Proof of 2). ($=$) Every $P \in \mathcal{P}_{n+1}$ can be written as $P(z) = 1 - zQ(z)$ with $\deg(Q) \le n-1$, and the computation in the claim shows $P(A)\,\varepsilon_0 = \bar{x} - Q(A)\,b$, where $Q(A)\,b$ ranges over all of $\mathcal{K}_n$. So the infimum is the minimum of $\|\bar{x} - x\|_A/\|\varepsilon_0\|_A$ over $x \in \mathcal{K}_n$, which by part 1) of Theorem 16.10 is attained at $x = x_n$.

($\le$) Write $\varepsilon_0 = \sum_j a_j v_j$, where $v_1, \ldots, v_N$ are orthonormal eigenvectors of A with eigenvalues $\lambda_1, \ldots, \lambda_N$. For every polynomial P,

$$P(A)\,\varepsilon_0 = \sum_j a_j P(\lambda_j)\,v_j.$$

Thus

$$\|\varepsilon_0\|_A^2 = \sum_j a_j^2\lambda_j, \qquad \|P(A)\,\varepsilon_0\|_A^2 = \sum_j a_j^2\lambda_j P(\lambda_j)^2,$$

and so

$$\frac{\|P(A)\,\varepsilon_0\|_A^2}{\|\varepsilon_0\|_A^2} = \frac{\sum_j a_j^2\lambda_j P(\lambda_j)^2}{\sum_j a_j^2\lambda_j} \le \max_{\lambda\in\lambda(A)} |P(\lambda)|^2.$$

Taking the infimum over $P \in \mathcal{P}_{n+1}$ gives the inequality. □

Theorem 16.14. *Let $\kappa = \lambda_{\max}(A)/\lambda_{\min}(A)$. For CG solving $Ax = b$, with A SPD and $b \neq 0$, and for every n such that x_n is defined,*

$$\frac{\|\varepsilon_n\|_A}{\|\varepsilon_0\|_A} \le 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^n.$$

Proof. For $n = 0$ the bound is $1 \le 2$. If $\kappa = 1$, then $A = \lambda_{\min}(A)\,I$ and $\bar{x} \in \mathcal{K}_1$, so $\varepsilon_n = 0$ for $n \ge 1$ by Theorem 16.10. So let $n \ge 1$ and $\kappa > 1$. Since x_n is defined, $g_0, \ldots, g_{n-1}$ are nonzero, and Theorem 16.12 gives

$$\frac{\|\varepsilon_n\|_A}{\|\varepsilon_0\|_A} \le \inf_{P'\in\mathcal{P}_{n+1}} \max_{\lambda\in\lambda(A)} |P'(\lambda)| \le \max_{\lambda\in\lambda(A)} |P(\lambda)|$$

for any $P(x)$ such that $\deg(P) \le n$ and $P(0) = 1$.

Recall that

$$m = \lambda_{\min} \le \lambda(A) \le \lambda_{\max} = M,$$

and that $G_\alpha = I - \alpha A$, with $\alpha = 2/(M+m)$ the optimal step size of Lecture 14, satisfies $-\gamma \le \lambda(G_\alpha) \le \gamma$ for

$$\gamma = \frac{M-m}{M+m} = \frac{\kappa - 1}{\kappa + 1}.$$

Let

$$\begin{aligned} P(x) &= T_n\Big(\gamma^{-1} - \frac{2x}{M-m}\Big)\Big/T_n(\gamma^{-1}) \qquad (T_n \text{ the Chebyshev polynomial}) \\ &= T_n\Big(\frac{M+m-2x}{M-m}\Big)\Big/T_n(\gamma^{-1}). \end{aligned}$$

Then $P(0) = 1$ and $\deg(P) = n$. For $m \le x \le M$, the argument $(M+m-2x)/(M-m)$ lies in $[-1, 1]$, where $|T_n| \le 1$ by the minimax theorem for Chebyshev polynomials in Lecture 14; hence

$$|P(x)| \le 1/T_n(\gamma^{-1}) \qquad \text{for } m \le x \le M.$$

By the bound $T_n(\gamma^{-1}) \ge \frac{1}{2}(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1})^n$ of Lecture 14, with the same $\gamma = \frac{\kappa-1}{\kappa+1}$, we obtain

$$|P(\lambda)| \le 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^n \qquad \text{for all } \lambda \in \lambda(A). \qquad \square$$

17 Preconditioned CG and Low-Stretch Spanning Trees

Lecture of February 26, 2020. Based on Gary L. Miller's handwritten notes.

Richardson's method and the conjugate gradient method (Lectures 14 to 16) need a number of iterations that grows with the condition number of the matrix, and the Laplacian of a path already has condition number of order n^2. Preconditioning replaces the system $Ax = b$ by an equivalent system with a better condition number, at the price of solving one system with a simpler matrix B in each iteration. This lecture expresses the relevant condition number through generalized eigenvalues and the Loewner order (Lecture 9), and then takes B to be the Laplacian of a subgraph, in particular of a spanning tree. We show that a maximum weight spanning tree gives condition number at most mn, and that this bound cannot be improved by more than a constant factor. Finally, the trace of $L_T^{\dagger} L_G$, a sum of effective resistances, gives the average generalized eigenvalue; on the cycle it shows that preconditioned CG can converge much faster than the condition number predicts.

Throughout this lecture, A and B are real symmetric $n \times n$ matrices. Graphs $G = (V, E, w)$ are undirected, with positive edge weights, $n \geq 2$ vertices and m edges, and L_G is the Laplacian of G. For graphs G and H on the same vertex set we write $G \preceq H$ when $L_G \preceq L_H$. We index iterates by k, so that n always denotes the dimension.

17.1 Preconditioned iterative methods

Our goal is to solve $Ax = b$, where A is symmetric positive definite (SPD) or symmetric positive semidefinite (SPSD). The basic idea for faster convergence is to choose an invertible matrix B, called a *preconditioner*, and to apply an iterative method to the system

$$B^{-1} A x = B^{-1} b.$$

Since B^{-1} is invertible, a vector $\bar{x}$ satisfies $A\bar{x} = b$ if and only if $B^{-1} A \bar{x} = B^{-1} b$, so the two systems have the same solutions. As an example, Richardson's iteration (Lecture 14) applied to the new system reads

$$x_{k+1} = x_k + (B^{-1} b - B^{-1} A x_k) = x_k + B^{-1}(b - A x_k).$$

It uses B^{-1} only through one linear system with matrix B per iteration.

Algorithm 17.1 (Preconditioned Richardson)**.** Input: A, B, b and the current iterate x_k.

1) Compute the residual $r = b - A x_k$.

2) Solve $B y = r$.

3) Return $x_{k+1} = x_k + y$.

Step 2 can be done by factoring $B = LU$ once in advance, or recursively, by another iterative method.

17.2 Examples of preconditioners

1) The *extrapolated method*: $B = \alpha^{-1} I$, where $\alpha = 2/(\lambda_{\min}(A) + \lambda_{\max}(A))$. Then step 3 of Algorithm 17.1 is $x_{k+1} = x_k + \alpha(b - Ax_k)$, the extrapolated method with the optimal step size of Lecture 14.

2) *Jacobi*: $B = \operatorname{diag}(A_{11}, \ldots, A_{nn})$, the diagonal part of A.

3) *Gauss–Seidel*: B is the lower triangular part of A, including the diagonal.

4) The symmetric successive over-relaxation (SSOR) preconditioner, which we do not discuss.

5) *Graph-theoretic* preconditioners, the subject of the rest of this lecture.

17.3 Convergence rates per iteration

For SPD matrices A and B, the eigenvalues of $B^{-1}A$ are real and positive (Section 17.4), and we set $\kappa(B^{-1}A) \coloneqq \lambda_{\max}(B^{-1}A)/\lambda_{\min}(B^{-1}A)$. Following Lecture 14, we say that an iterative method *converges at rate* $r \in (0, 1]$ if there is a constant c, independent of k, such that after k iterations the error, measured in a fixed norm, is at most $c(1-r)^k$ times the initial error. Since $(1-r)^k \le e^{-rk}$, each further $\lceil \ln 2/r \rceil$ iterations halve this bound; we say that the method needs $O(1/r)$ *iterations per bit* of accuracy.

Preconditioned CG (PCG) is the conjugate gradient method applied to the SPD system $\tilde{A}y = \tilde{b}$, where $\tilde{A} = B^{-1/2}AB^{-1/2}$ and $\tilde{b} = B^{-1/2}b$, followed by $x = B^{-1/2}y$. It can be organized so that each iteration multiplies one vector by A and solves one system with matrix B, without computing $B^{-1/2}$; we omit the details.

Theorem 17.2. *Let A and B be SPD, let $\kappa = \kappa(B^{-1}A)$, let $\bar{x} = A^{-1}b$, and let x_k denote the k-th iterate.*

1) Preconditioned Richardson, run with the preconditioner $\gamma^{-1}B$, where $\gamma = 2/(\lambda_{\min}(B^{-1}A) + \lambda_{\max}(B^{-1}A))$, satisfies

$$\|\bar{x} - x_k\|_A \le \Big(\frac{\kappa - 1}{\kappa + 1}\Big)^k \|\bar{x} - x_0\|_A.$$

So it converges at rate $2/(\kappa+1) \ge 1/\kappa$ and needs $O(\kappa)$ iterations per bit.

2) PCG started at $x_0 = 0$ satisfies

$$\|\bar{x} - x_k\|_A \le 2\Big(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\Big)^k \|\bar{x}\|_A.$$

So it converges at rate $2/(\sqrt{\kappa}+1) \ge 1/\sqrt{\kappa}$ and needs $O(\sqrt{\kappa})$ iterations per bit.

Proof. Let $\lambda_- = \lambda_{\min}(B^{-1}A)$ and $\lambda_+ = \lambda_{\max}(B^{-1}A)$, so that $\kappa = \lambda_+/\lambda_-$. The matrix $S = A^{1/2}B^{-1}A^{1/2}$ is symmetric, and $B^{-1}A = A^{-1/2}SA^{1/2}$, so S has the same eigenvalues as $B^{-1}A$; they lie in $[\lambda_-, \lambda_+]$. Similarly $\tilde{A}$ is symmetric and $B^{-1}A = B^{-1/2}\tilde{A}B^{1/2}$, so $\tilde{A}$ has the same eigenvalues and $\kappa(\tilde{A}) = \kappa$.

1) Write $e_k = \bar{x} - x_k$. Since $b = A\bar{x}$, the iteration $x_{k+1} = x_k + \gamma B^{-1}(b - Ax_k)$ gives $e_{k+1} = (I - \gamma B^{-1}A)e_k$, and hence $A^{1/2}e_{k+1} = (I - \gamma S)A^{1/2}e_k$. The eigenvalues of the symmetric matrix $I - \gamma S$ lie in $[1 - \gamma\lambda_+, 1 - \gamma\lambda_-]$, and with $\gamma = 2/(\lambda_- + \lambda_+)$ both endpoints have absolute value $(\lambda_+ - \lambda_-)/(\lambda_+ + \lambda_-) = (\kappa - 1)/(\kappa + 1)$. Since $\|e\|_A = \|A^{1/2}e\|_2$, each iteration multiplies

the A-norm of the error by at most $(\kappa-1)/(\kappa+1) = 1 - 2/(\kappa+1)$. Finally $2/(\kappa+1) \geq 1/\kappa$ because $\kappa \geq 1$.

2) Let y_k be the CG iterates for $\tilde{A}y = \tilde{b}$ with $y_0 = 0$, and $\bar{y} = B^{1/2}\bar{x}$ its solution, so that $x_k = B^{-1/2}y_k$. For every y, $\|y\|_{\tilde{A}}^2 = (B^{-1/2}y)^\top A(B^{-1/2}y)$, so $\|\bar{y} - y_k\|_{\tilde{A}} = \|\bar{x} - x_k\|_A$ and $\|\bar{y}\|_{\tilde{A}} = \|\bar{x}\|_A$. The bound now follows from the convergence theorem for CG (Theorem 16.14) applied to $\tilde{A}$, and $2/(\sqrt{\kappa}+1) \geq 1/\sqrt{\kappa}$. □

Our goal is therefore to find a preconditioner B such that

1) $\kappa(B^{-1}A)$ is small, which gives a better convergence rate, and

2) solving $By = r$ is not too expensive.

17.4 Estimating $\kappa(B^{-1}A)$

If B is invertible, then $B^{-1}Ax = \lambda x$ if and only if $Ax = \lambda Bx$.

Definition 17.3. Let A and B be $n \times n$ matrices with B invertible. A number λ and a vector $x \neq 0$ with $Ax = \lambda Bx$ are a *generalized eigenvalue* and a *generalized eigenvector* of the pair (A, B). They are the eigenvalues and eigenvectors of $B^{-1}A$.

Laplacians are singular, so we also need a version of this definition for singular matrices with the same null space.

Definition 17.4. Let A and B be nonzero SPSD matrices with $\mathrm{Null}(A) = \mathrm{Null}(B)$. Define

$$\lambda_f(A, B) \coloneqq \{\lambda : Ax = \lambda Bx \text{ for some } x \notin \mathrm{Null}(A)\}.$$

When A and B are invertible, $\lambda_f(A, B)$ is the set of generalized eigenvalues of (A, B). For $\lambda \neq 0$ we have $Ax = \lambda Bx$ if and only if $Bx = \lambda^{-1}Ax$. If B is SPD, then

$$Ax = \lambda Bx \quad \text{if and only if} \quad B^{-1/2}AB^{-1/2}(B^{1/2}x) = \lambda(B^{1/2}x),$$

as we see by multiplying on the left by $B^{-1/2}$. The matrix $B^{-1/2}AB^{-1/2}$ is symmetric, so every generalized eigenvalue is real; if A is also SPD, this matrix is SPD and every generalized eigenvalue is positive. The same holds in the setting of Definition 17.4. Let N be the common null space. If $x \notin N$, write $x = x_N + x'$ with $x_N \in N$ and $0 \neq x' \in N^\perp$. Then $x^\top Ax = x'^\top Ax' > 0$ and $x^\top Bx = x'^\top Bx' > 0$, since a vector z with $z^\top Az = 0$ satisfies $Az = 0$ for SPSD A. Hence $Ax = \lambda Bx$ gives $\lambda = x^\top Ax / x^\top Bx > 0$.

Claim 17.5. *Let A and B be as in Definition 17.4. Then all elements of $\lambda_f(A, B)$ are positive, $\lambda_f(B, A) = \{1/\lambda : \lambda \in \lambda_f(A, B)\}$, and*

$$\kappa(B^{-1}A) \coloneqq \frac{\max \lambda_f(A, B)}{\min \lambda_f(A, B)} = \max \lambda_f(A, B) \max \lambda_f(B, A).$$

When B is invertible, this agrees with $\lambda_{\max}(B^{-1}A)/\lambda_{\min}(B^{-1}A)$.

Proof. Positivity was shown above. Since $0 \notin \lambda_f(A, B)$, and since $Ax = \lambda Bx$ holds if and only if $Bx = \lambda^{-1}Ax$ for $\lambda \neq 0$, we obtain the description of $\lambda_f(B, A)$. Hence $\max \lambda_f(B, A) = 1/\min \lambda_f(A, B)$. □

Definition 17.6. For SPSD matrices A and B, let $\sigma(A/B) \coloneqq \min\{\tau \in \mathbb{R} : A \preceq \tau B\}$.

Lemma 17.7. *If A and B are as in Definition 17.4, then the minimum in Definition 17.6 exists and $\sigma(A/B) = \max \lambda_f(A, B)$.*

Proof. Let N be the common null space, and let U be a matrix whose columns form an orthonormal basis of $N^\perp$. For every x, the values $x^\top Ax$ and $x^\top Bx$ depend only on the component of x in $N^\perp$, as shown above. Hence $A \preceq \tau B$ if and only if $u^\top A' u \le \tau\, u^\top B' u$ for all u, where $A' = U^\top AU$ and $B' = U^\top BU$. Both are SPD. With $u = B'^{-1/2} z$, this condition says that τ is at least every Rayleigh quotient of the symmetric matrix $C = B'^{-1/2} A' B'^{-1/2}$, so the least such τ exists and equals $\lambda_{\max}(C)$. Finally, $\lambda_f(A, B)$ is the set of eigenvalues of C. Indeed, for $x = Uu$ the vectors Ax and Bx lie in $N^\perp$, the range of U, so $Ax = \lambda Bx$ if and only if $A'u = \lambda B'u$, that is, $C(B'^{1/2}u) = \lambda(B'^{1/2}u)$; and every $x \notin N$ may be replaced by its component $Uu \neq 0$ in $N^\perp$. □

Combining Claim 17.5 with Lemma 17.7, applied to (A, B) and to (B, A), we obtain

$$\kappa(B^{-1}A) = \sigma(A/B) \cdot \sigma(B/A). \tag{17.1}$$

17.5 Subgraph preconditioners

Let H be a connected spanning subgraph of a connected graph G: it has the same vertex set, a subset of the edges, and the same weights on them. Set $A = L_G$ and $B = L_H$. Both null spaces equal $\operatorname{span}\{\mathbf{1}\}$, since both graphs are connected. For graphs we write

$$\kappa(G, H) \coloneqq \kappa(L_H^{-1} L_G) \quad \text{and} \quad \sigma(G/H) \coloneqq \sigma(L_G/L_H),$$

where $\kappa(L_H^{-1} L_G)$ is defined through $\lambda_f(L_G, L_H)$ as in Claim 17.5.

Proposition 17.8. *Let H be a connected spanning subgraph of a connected graph G, and let $A = L_G$ and $B = L_H$. Then $B \preceq A$, every $\lambda \in \lambda_f(A, B)$ satisfies $\lambda \ge 1$, and*

$$\kappa(B^{-1}A) \le \sigma(A/B) = \min\{\tau : A \preceq \tau B\}.$$

Proof. The difference $L_G - L_H$ is the Laplacian of the graph formed by the edges of G not in H, so it is positive semidefinite, and $B \preceq A$. Thus $\sigma(B/A) \le 1$, so every $\lambda \in \lambda_f(A, B)$ satisfies $\lambda \ge 1$ by Claim 17.5 and Lemma 17.7, and $\kappa(B^{-1}A) = \sigma(A/B)\,\sigma(B/A) \le \sigma(A/B)$ by (17.1). □

So every subgraph preconditioner has condition number at most $\sigma(A/B)$.

Remark 17.9. Theorem 17.2 applies to $A = L_G$ and $B = L_H$ for connected graphs G and H on the same vertex set. Both matrices map $\mathbf{1}^\perp$ to itself and are positive definite there, and $L_G x = b$ is solvable exactly when $b \perp \mathbf{1}$. If also $x_0 \perp \mathbf{1}$, and B^{-1} and $B^{-1/2}$ are read as $B^\dagger$ and $(B^\dagger)^{1/2}$, then all iterates lie in $\mathbf{1}^\perp$, and the theorem holds there with $\lambda_{\min}(B^{-1}A)$ and $\lambda_{\max}(B^{-1}A)$ read as $\min \lambda_f(A, B)$ and $\max \lambda_f(A, B)$, and with $\kappa(B^{-1}A)$ as in Claim 17.5.

17.6 Spanning tree preconditioners

The advantage of a spanning tree T as a preconditioner is that a system $L_T y = r$ with $r \perp \mathbf{1}$ can be solved in $O(n)$ time: trees have perfect elimination orders (Lecture 13), so Gaussian

elimination that removes leaves one at a time creates no fill. Vaidya [Vai90] proposed to take T to be a maximum weight spanning tree of G. We bound

$$\sigma(G/T) = \min_{\tau}\{L_G \preceq \tau L_T\},$$

which is an upper bound on $\kappa(G,T)$ by Proposition 17.8, with the path embedding argument of Lecture 9.

Proposition 17.10. *Let G be a connected graph with n vertices, m edges and positive weights, and let T be a maximum weight spanning tree of G. Then $L_G \preceq mn\, L_T$, and hence $\kappa(G,T) \le \sigma(G/T) \le mn$.*

Proof. Embed G into T by mapping each edge $e = (u,v)$ of G to the unique path P_e from u to v in T. Each edge of T lies on at most m of these paths, so the congestion is at most m, and each path has at most $n-1$ edges, so the dilation is at most $n-1 \le n$. For unit weights, the embedding theorem (Theorem 9.22) gives $L_G \preceq C \cdot D\, L_T \preceq mn\, L_T$.

For general weights we use the maximality of T. If f is an edge of P_e, then $w(f) \ge w(e)$: removing f from T leaves two components, with u and v in different ones, so adding e gives a spanning tree, whose weight would exceed that of T if $w(e) > w(f)$. Let $L_e = \chi_{uv}\chi_{uv}^{\top}$ be the Laplacian of the edge e with unit weight. The Cauchy–Schwarz step in the proof of Theorem 9.22 shows that $L_e \preceq |P_e| \sum_{f \in P_e} L_f$. Hence

$$w(e) L_e \preceq (n-1) \sum_{f \in P_e} w(e) L_f \preceq (n-1) \sum_{f \in P_e} w(f) L_f.$$

Summing over $e \in E$, and using that each edge f of T lies on at most m paths P_e and that $L_f \succeq 0$, we get $L_G = \sum_e w(e) L_e \preceq (n-1)\, m \sum_{f \in T} w(f) L_f = (n-1)\, m\, L_T$. The bound on $\kappa(G,T)$ follows from Proposition 17.8. □

The bound mn is the best possible up to a constant factor, even for maximum weight spanning trees.

Example 17.11. Let $n \ge 4$ be even and $k = n/2$. Let G be the *ladder* with vertices $b_1, \dots, b_k$ and $t_1, \dots, t_k$, the edges $b_i b_{i+1}$ and $t_i t_{i+1}$ for $1 \le i < k$, and the rungs $b_i t_i$ for $1 \le i \le k$, all of weight 1; so $m = 3k - 2 < 3n/2$. Let T consist of the two paths and the rung $b_1 t_1$ (Fig. 4). All weights are equal, so T is a maximum weight spanning tree. Take the test vector $x(b_i) = i$ and $x(t_i) = -i$. Each of the $2(k-1)$ path edges contributes 1 to $x^{\top} L_T x$, and the rung $b_1 t_1$ contributes $2^2 = 4$, so $x^{\top} L_T x = n + 2$. Each other rung $b_i t_i$ contributes $(2i)^2$ to $x^{\top} L_G x$, so

$$x^{\top} L_G x = n + 2 + \sum_{i=2}^{k} 4i^2 = \frac{n(n+1)(n+2)}{6} + n - 2.$$

Hence $x^{\top} L_T x \approx n$ and $x^{\top} L_G x \approx n^3/6$, and

$$\sigma(G/T) \ge \frac{x^{\top} L_G x}{x^{\top} L_T x} > \frac{n(n+1)}{6}.$$

Together with Proposition 17.10, this gives $\sigma(G/T) = \Theta(n^2) = \Theta(mn)$. Moreover $\kappa(G,T) = \sigma(G/T)$ here: the vector y with $y(b_1) = 1$, $y(t_1) = -1$ and $y = 0$ elsewhere has $y^{\top} L_G y = y^{\top} L_T y = 6$, so $\sigma(T/G) \ge 1$, while $\sigma(T/G) \le 1$ because $T \preceq G$.

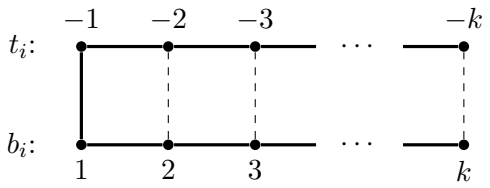


Figure 4: The ladder G and its spanning tree T of Example 17.11. Solid edges form T; dashed rungs are the edges of G not in T. Each vertex is labeled with its value under the test vector x.

Theorem 17.12. *Let G be a connected graph with n vertices, m edges and positive weights. PCG with a maximum weight spanning tree preconditioner solves $L_G x = b$, for $b \perp \mathbf{1}$, in $O(m^{3/2}n^{1/2})$ time per bit:*

1) $O(m \log n)$ time to find the tree;

2) $O(m)$ time per iteration;

3) $O((mn)^{1/2})$ iterations per bit.

Proof. 1) Kruskal's algorithm finds a maximum weight spanning tree in $O(m \log n)$ time. The randomized algorithm of Karger, Klein and Tarjan [KKT95], applied to the negated weights, finds one in expected $O(m)$ time. 2) Each iteration multiplies one vector by L_G, in $O(m)$ time, solves one system with L_T, in $O(n)$ time, and performs $O(n)$ further vector operations; and $n \le m + 1$. 3) By Proposition 17.10, $\kappa(G, T) \le mn$, so Theorem 17.2 with Remark 17.9 gives $O(\sqrt{mn})$ iterations per bit. The product of 2) and 3) is $O(m^{3/2}n^{1/2})$, and the one-time cost $O(m \log n)$ is within this bound since $m \ge n - 1$. □

17.7 Finding a better spanning tree

A spanning tree could improve on Proposition 17.10 in two ways: it could have a better condition number, or a better distribution of its generalized eigenvalues. The second way matters because of the following property of CG, proved in Lecture 16.

Fact 17.13. *Let A be SPD, $\bar{x} = A^{-1}b$, and let x_k be the k-th CG iterate for $Ax = b$ started at $x_0 = 0$. For every real polynomial P with $\deg P \le k$ and $P(0) = 1$,*

$$\|\bar{x} - x_k\|_A \le \max_{\lambda \in \lambda(A)} |P(\lambda)| \, \|\bar{x}\|_A.$$

So CG converges fast when some polynomial of low degree with $P(0) = 1$ is small at all eigenvalues, which can happen even when the eigenvalues are spread over a wide range.

Example 17.14. Let $n \ge 3$, let $G = C_n$ be the cycle with unit weights, and let $T = P_n$ be the spanning tree obtained by deleting the edge $e = (1, n)$.

Claim 17.15. *$\kappa(C_n, P_n) = \Theta(n)$; more precisely, $n \le \kappa(C_n, P_n) \le 2(n-1)$.*

Proof. For the upper bound, $P_n \preceq C_n$ gives $\sigma(P_n/C_n) \le 1$, and the vector $x = \chi_1 + \chi_n$ has $x^\top L_{P_n} x = x^\top L_{C_n} x = 2$, so $\sigma(P_n/C_n) = 1$. Embedding each edge of P_n into itself and e into

the whole path gives congestion 2 and dilation $n-1$, so $\sigma(C_n/P_n) \le C \cdot D = 2(n-1)$ by Theorem 9.22. For the lower bound, take $x_i = i$. Then

$$\frac{x^\top L_{C_n} x}{x^\top L_{P_n} x} = \frac{(n-1)^2 + (n-1)}{n-1} = n,$$

so $\sigma(C_n/P_n) \ge n$, and $\kappa(C_n, P_n) = \sigma(C_n/P_n)\,\sigma(P_n/C_n) \ge n$ by (17.1). □

So the largest generalized eigenvalue of (C_n, P_n) is $\sigma(C_n/P_n) \ge n$ (Homework 17.19 shows that it equals n), and the smallest is $1/\sigma(P_n/C_n) = 1$. To see how the $n-1$ generalized eigenvalues are distributed between 1 and n, we compute their average, that is, their sum $\mathrm{Tr}(P_n^\dagger C_n)$ divided by $n-1$.

17.8 The trace and effective resistances

We use three properties of the trace.

Fact 17.16.

1) $\mathrm{Tr}(\alpha A + B) = \alpha \,\mathrm{Tr}(A) + \mathrm{Tr}(B)$ *for square* A, B *of the same size and* $\alpha \in \mathbb{R}$.

2) $\mathrm{Tr}(A) = \sum_{i=1}^n A_{ii} = \sum_{i=1}^n \lambda_i$, *where* $\lambda_1, \ldots, \lambda_n$ *are the eigenvalues of the* $n \times n$ *matrix* A, *with multiplicity.*

3) $\mathrm{Tr}(ABC) = \mathrm{Tr}(CAB)$ *whenever the products are defined, also for rectangular matrices.*

Let G and H be graphs on the same vertex set V, with H connected, and let $A = L_G$ and $B = L_H$; in our application $H \subsetneq G$ is a spanning tree. For $u, v \in V$, let $\chi_{uv} = \chi_u - \chi_v$, the vector with 1 at u, -1 at v and 0 elsewhere.

Definition 17.17. For $u, v \in V$, $\mathrm{ER}_B(u, v)$ is the effective resistance between u and v in the graph H with Laplacian B (Lectures 2 and 3), where an edge of weight w has resistance $1/w$. For an edge $e = (u, v)$ we write $\mathrm{ER}_B(e) = \mathrm{ER}_B(u, v)$.

By Proposition 3.11, $\mathrm{ER}_B(u, v) = V_v - V_u$ for every vector V with $BV = \chi_v - \chi_u$. Since $\chi_v - \chi_u \perp \mathbf{1}$ and $BB^\dagger$ is the orthogonal projection onto $\mathbf{1}^\perp$, the vector $V = B^\dagger(\chi_v - \chi_u)$ is such a vector, and therefore

$$\mathrm{ER}_B(u, v) = \chi_{uv}^\top B^\dagger \chi_{uv}.$$

Claim 17.18. *With* G, H, A *and* B *as above,*

$$\mathrm{Tr}(B^\dagger A) = \sum_{e \in E(G)} w_G(e)\ \mathrm{ER}_B(e).$$

Proof. For an edge (u, v), let $L_{uv} = \chi_{uv}\chi_{uv}^\top$, the Laplacian of the single edge (u, v) with weight 1:

$$\chi_{uv} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ -1 \\ 0 \end{pmatrix} \begin{matrix} \\ \\ \\ \leftarrow u \\ \\ \leftarrow v \\ \\ \end{matrix}$$

Then $A = \sum_{(u,v)\in E(G)} w_G(u,v) L_{uv}$ (Proposition 9.19), and

$$\begin{aligned}
\operatorname{Tr}(B^\dagger A) &= \operatorname{Tr}\Big(\sum_{(u,v)\in E(G)} w_G(u,v)\, B^\dagger \chi_{uv} \chi_{uv}^\top \Big) && \text{by the definition of } L_{uv} \\
&= \sum_{(u,v)\in E(G)} w_G(u,v) \operatorname{Tr}(B^\dagger \chi_{uv} \chi_{uv}^\top) && \text{by Fact 17.16, part 1} \\
&= \sum_{(u,v)\in E(G)} w_G(u,v) \operatorname{Tr}(\chi_{uv}^\top B^\dagger \chi_{uv}) && \text{by Fact 17.16, part 3} \\
&= \sum_{(u,v)\in E(G)} w_G(u,v)\, \chi_{uv}^\top B^\dagger \chi_{uv} && \text{the trace of a } 1\times 1 \text{ matrix} \\
&= \sum_{(u,v)\in E(G)} w_G(u,v)\, \operatorname{ER}_B(u,v) && \text{by the formula above.}
\end{aligned}$$

□

17.9 Back to C_n with tree P_n

The eigenvalues of $P_n^\dagger C_n$ (writing C_n and P_n for their Laplacians) are 0, with eigenvector $\mathbf{1}$, and the $n-1$ elements of $\lambda_f(C_n, P_n)$ with multiplicity. Indeed, $P_n^\dagger C_n \mathbf{1} = 0$, the range of $P_n^\dagger$ is $\mathbf{1}^\perp$, and for $x \perp \mathbf{1}$ we have $P_n^\dagger C_n x = \lambda x$ if and only if $C_n x = \lambda P_n x$ (multiply by P_n, respectively by $P_n^\dagger$). By Fact 17.16 and Claim 17.18,

$$\sum_{\lambda_i \in \lambda_f(C_n,P_n)} \lambda_i = \operatorname{Tr}(P_n^\dagger C_n) = \sum_{e\in C_n} \operatorname{ER}_T(e),$$

where ER_T is the effective resistance in $T = P_n$. In a tree, the effective resistance between two vertices is the sum of the resistances along the path joining them (resistors in series, Homework 2.1). So $\operatorname{ER}_T(e) = 1$ for each of the $n-1$ edges e of P_n, and $\operatorname{ER}_T(e) = n-1$ for the edge $e = (1,n) \notin P_n$. Thus

$$\sum_{\lambda_i \in \lambda_f(C_n,P_n)} \lambda_i = 2(n-1),$$

and the average generalized eigenvalue is 2, although the largest one is at least n.

Homework 17.19. Show that the eigenvalues of $P_n^\dagger C_n$ are 0 once, 1 with multiplicity $n-2$, and n once.

Consider the polynomial

$$P(z) = (1-z)\Big(1 - \frac{z}{n}\Big).$$

By Homework 17.19, P vanishes at every element of $\lambda_f(C_n, P_n)$, so $P(P_n^\dagger C_n)\, x = 0$ for every $x \perp \mathbf{1}$. Since $\deg P = 2$ and $P(0) = 1$, Fact 17.13, applied to the preconditioned system on $\mathbf{1}^\perp$ (Remark 17.9), shows that PCG on (C_n, P_n) reaches the exact solution after at most 2 iterations, for every n, although $\kappa(C_n, P_n) \geq n$.

18 Preconditioned CG and LSST Continued

Lecture of March 4, 2020. Based on Gary L. Miller's handwritten notes.

In Lecture 17 we saw that the trace of $L_T^\dagger L_G$, the sum of the generalized eigenvalues of (L_G, L_T), is a sum of effective resistances, and that PCG can be fast when only a few generalized eigenvalues are large. This lecture identifies that trace with the total stretch of the edges of G over the tree T, and quotes the existence of low-stretch spanning trees, whose total stretch is $O(m \log n (\log\log n)^2)$. We then show that some polynomial of low degree is small at all eigenvalues when only a few eigenvalues are large, and conclude that PCG with a low-stretch spanning tree as preconditioner needs $O(m^{1/3} \log n)$ iterations per bit.

Throughout this lecture, $G = (V, E, w)$ is a connected undirected graph with positive edge weights, $n \geq 2$ vertices and m edges, and T is a spanning tree of G. We use the notation of Lecture 17: the set λ_f of generalized eigenvalues (Definition 17.4), preconditioned CG (PCG), and iterations per bit.

18.1 Low stretch spanning trees (LSST)

Definition 18.1. Let $e = (u, v) \in E$, and let $w_1, \ldots, w_k$ be the weights of the edges on the path from u to v in T. The *stretch* of e with respect to T is

$$\operatorname{stretch}_T(e) \coloneqq w_G(e) \sum_{i=1}^{k} \frac{1}{w_i}.$$

An edge of T has stretch 1. If each edge of weight w is a resistor of resistance $1/w$, then $\sum_i 1/w_i$ is the effective resistance $\operatorname{ER}_T(u, v)$ between u and v in T, by the series rule (Homework 2.1). Hence $\operatorname{stretch}_T(e) = w_G(e) \operatorname{ER}_T(e)$, and Claim 17.18 with $H = T$ gives

$$\operatorname{Tr}(L_T^\dagger L_G) = \sum_{e \in E} \operatorname{stretch}_T(e). \tag{18.1}$$

Theorem 18.2. *For every connected graph G with positive edge weights, n vertices and m edges, there exists a spanning tree T of G such that*

$$\sum_{e \in E} \operatorname{stretch}_T(e) = O(m \log n (\log\log n)^2).$$

We call such a tree a *low-stretch spanning tree* (LSST). We state Theorem 18.2 without proof: both the algorithm that finds T and its analysis are nontrivial.

18.2 Polynomials that are small at all but a few eigenvalues

The following theorem refines the Chebyshev bound behind the convergence of CG (Theorem 16.14): the k numbers above β cost only k extra degrees, and the remaining numbers enter only through the ratio β/α.

Theorem 18.3 (Spielman–Woo [SW09])**.** *Let $0 < \alpha \leq \beta$, let $0 \leq k \leq N$ be integers, and let $\lambda_1, \ldots, \lambda_N$ be real numbers with*

$$\lambda_1 \geq \cdots \geq \lambda_k \geq \beta \geq \lambda_{k+1} \geq \cdots \geq \lambda_N \geq \alpha.$$

Then for every integer $t \geq k$ there exists a polynomial P of degree at most t such that

1) $P(0) = 1$, *and*

2) for every $i \in \{1, \dots, N\}$, $|P(\lambda_i)| \le 2\left(\dfrac{\sqrt{\beta/\alpha}-1}{\sqrt{\beta/\alpha}+1}\right)^{t-k}$.

Proof. Let $d = t - k$, and let r be a polynomial of degree at most d with $r(0) = 1$ such that

$$|r(x)| \le 2\left(\frac{\sqrt{\beta/\alpha}-1}{\sqrt{\beta/\alpha}+1}\right)^{d} \qquad \text{for all } \alpha \le x \le \beta.$$

For $\alpha < \beta$, the scaled Chebyshev polynomial of degree d from the proof of Theorem 16.14, built for the interval $[\alpha, \beta]$ in place of $[\lambda_{\min}, \lambda_{\max}]$, has these properties. For $\alpha = \beta$ the right side is 2 if $d = 0$ and 0 if $d \ge 1$, and $r(x) = (1 - x/\alpha)^d$ works. Set

$$P(x) = r(x) \prod_{i:\ \lambda_i > \beta} \Big(1 - \frac{x}{\lambda_i}\Big).$$

Since $\lambda_{k+1} \le \beta$, only indices $i \le k$ can have $\lambda_i > \beta$, so the product has at most k factors and $\deg P \le d + k = t$. Also $P(0) = r(0) = 1$. It remains to bound $|P(\lambda_i)|$ for $1 \le i \le N$.

Case $\lambda_i > \beta$. The factor $1 - x/\lambda_i$ vanishes at $x = \lambda_i$, so $P(\lambda_i) = 0$.

Case $\lambda_i \le \beta$. Then $\alpha \le \lambda_i \le \beta$, so $|r(\lambda_i)|$ satisfies the bound. For each j with $\lambda_j > \beta$ we have $0 < \lambda_i/\lambda_j < 1$, so $0 < 1 - \lambda_i/\lambda_j < 1$. Hence $|P(\lambda_i)| \le |r(\lambda_i)|$, which proves the bound. □

18.3 PCG with a low-stretch spanning tree

Consider PCG with an LSST T as preconditioner. Let $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_{n-1}$ be the elements of $\lambda_f(L_G, L_T)$, listed with multiplicity and in decreasing order, as in Theorem 18.3; together with 0 they are the eigenvalues of $L_T^\dagger L_G$ (Section 17.9). Since $T \preceq G$, every λ_i satisfies $\lambda_i \ge 1$ (Proposition 17.8). By Fact 17.16, (18.1) and Theorem 18.2,

$$\sum_{i=1}^{n-1} \lambda_i = \operatorname{Tr}(L_T^\dagger L_G) = O(m \log n\, (\log\log n)^2).$$

Since the λ_i are positive and add up to this trace, most of them must be small: for every $\beta > 0$, at most $\operatorname{Tr}(L_T^\dagger L_G)/\beta$ of them are at least β.

Theorem 18.4. *Let T be a low-stretch spanning tree of G, as in Theorem 18.2. PCG with preconditioner L_T, applied to $L_G x = b$ with $b \perp \mathbf{1}$ and started at $x_0 = 0$, needs $O(m^{1/3} \log n)$ iterations per bit. More precisely, with $s = \operatorname{Tr}(L_T^\dagger L_G)^{1/3}$, after $6j\lceil s \rceil$ iterations the L_G-norm of the error is at most 2^{-j} times its initial value, for every integer $j \ge 0$.*

Proof. By Fact 17.13, applied to the preconditioned system on $\mathbf{1}^\perp$ (Remark 17.9), it suffices to find a polynomial P with $P(0) = 1$ and $\deg P \le 6\lceil s \rceil$ such that $|P(\lambda_i)| \le 1/2$ for every i. Then P^j has degree at most $6j\lceil s \rceil$, satisfies $P^j(0) = 1$, and has $|P^j(\lambda_i)| \le 2^{-j}$ for every i.

Set

$$\beta = \operatorname{Tr}(L_T^\dagger L_G)^{2/3} = s^2, \qquad k = \operatorname{Tr}(L_T^\dagger L_G)^{1/3} = s = \sqrt{\beta}.$$

Since each of the $n - 1 \ge 1$ numbers λ_i is at least 1, we have $s \ge 1$ and $\beta \ge 1$. To apply Theorem 18.3, let k' be the number of eigenvalues $\lambda_i \ge \beta$. Then $k'\beta \le \operatorname{Tr}(L_T^\dagger L_G)$, so $k' \le$

$\mathrm{Tr}(L_T^\dagger L_G)^{1/3} = \sqrt{\beta} = k$. Every λ_i is at least 1, so we take $\alpha = 1 \le \beta$, and we apply the theorem with k' in place of k and with $t = 2\lceil k \rceil \ge k'$. This gives a polynomial Q of degree at most $2\lceil k \rceil$ with $Q(0) = 1$. Since $t - k' \ge 2\lceil k \rceil - k \ge \sqrt{\beta}$ and $0 \le (\sqrt{\beta} - 1)/(\sqrt{\beta} + 1) < 1$,

$$|Q(\lambda_i)| \le 2\Big(\frac{\sqrt{\beta} - 1}{\sqrt{\beta} + 1}\Big)^{t - k'} \le 2\Big(1 - \frac{1}{\sqrt{\beta}}\Big)^{\sqrt{\beta}} \le \frac{2}{e}$$

for every i. Here we used $(\sqrt{\beta} - 1)/(\sqrt{\beta} + 1) \le (\sqrt{\beta} - 1)/\sqrt{\beta}$ and $1 - y \le e^{-y}$. The polynomial $P = Q^3$ has degree at most $6\lceil k \rceil = 6\lceil s \rceil$, satisfies $P(0) = 1$, and $|P(\lambda_i)| \le 8/e^3 < 1/2$ for every i.

Thus PCG needs $O(\mathrm{Tr}(L_T^\dagger L_G)^{1/3})$ iterations per bit. By Theorem 18.2, $\mathrm{Tr}(L_T^\dagger L_G)^{1/3} = O\big(m^{1/3} (\log n)^{1/3} (\log \log n)^{2/3}\big)$, which is $O(m^{1/3} \log n)$. □

19 Fiedler's Theorem (Nodal Domains) and Generalized Laplacians

Lecture of March 18, 2020. Based on Gary L. Miller's handwritten notes.

Eigenvectors of Laplacians appear throughout engineering, the sciences, economics, mathematics and computer science, so we want to understand their structure. So far we know only that the first eigenvector has one sign and that all the others take both signs. This lecture studies the sign patterns of the others through their nodal domains, the connected pieces of the graph on which an eigenvector keeps one sign. The main result is Fiedler's theorem: for $k \geq 2$, the k-th eigenvector has at most $k-1$ weak nodal domains, while the star shows that no such bound holds for nodal domains. We prove Fiedler's theorem for generalized Laplacians, using the Perron–Frobenius theorem (Lectures 5 and 6) and Cauchy's interlacing theorem, which we derive from the Courant–Fischer theorem (Lecture 9). Finally we show that an eigenvector of λ_2 with minimal support has a connected positive part and a connected negative part.

19.1 Fiedler's theorem on nodal domains

Throughout this section, $G = (V, E, w)$ is a connected undirected graph with $n \geq 2$ vertices, positive edge weights and positive vertex masses $m(v)$, L_G is its Laplacian, and $M = \operatorname{diag}(m(v))_{v \in V}$ is the mass matrix. We consider the generalized eigenvalue problem

$$L_G x = \lambda M x,$$

or equivalently, with $y = M^{1/2} x$,

$$M^{-1/2} L_G M^{-1/2} y = \lambda y.$$

The matrix $M^{-1/2} L_G M^{-1/2}$ is symmetric and positive semidefinite, and its kernel is spanned by $M^{1/2}\mathbf{1}$ because G is connected. Hence the eigenvalues are

$$0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_n,$$

and there are eigenvectors $v_1, v_2, \ldots, v_n$ of $L_G x = \lambda M x$ with $v_1 = \mathbf{1}$ and $v_i^{\top} M v_j = 0$ for $i \neq j$. We view each v_i as a function $v_i \colon V \to \mathbb{R}$.

So far we know only that every eigenvector of $\lambda_1 = 0$ is a nonzero multiple of $\mathbf{1}$, so it is strictly positive or strictly negative. For $k \geq 2$, the condition $v_k^{\top} M v_1 = \sum_v m(v) v_k(v) = 0$ with positive masses shows that v_k takes both positive and negative values. The goal of this lecture is to understand more of the structure of the eigenvectors $v_1, \ldots, v_n$.

Definition 19.1. For $f \colon V \to \mathbb{R}$, let

$$f^+ = \{v \in V \mid f(v) > 0\}, \qquad f^- = \{v \in V \mid f(v) < 0\}.$$

19.2 Nodal domains

Definition 19.2. The *induced graph* on $W \subseteq V$ is

$$G(W) = (W, E_W), \qquad E_W = \{(u, v) \in E \mid u, v \in W\}.$$

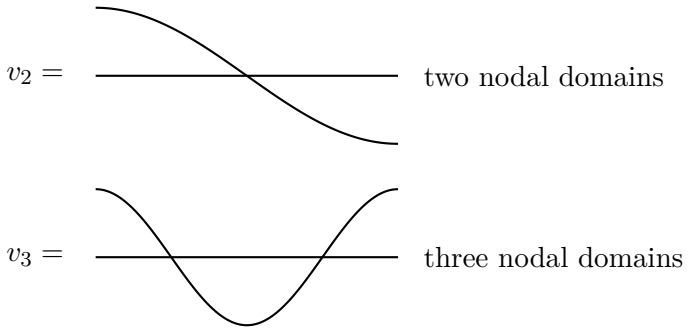
Definition 19.3. A *nodal domain* of $f : V \to \mathbb{R}$ is a connected component of $G(f^+)$ or of $G(f^-)$.

Definition 19.4. For $f : V \to \mathbb{R}$, let $W_f^+ = f^{+=} := \{v \in V \mid f(v) \geq 0\}$.

Definition 19.5. The *weak nodal domains* of f are the connected components of $G(f^{+=})$.

We could also use $f^{-=} = \{v \mid f(v) \leq 0\}$. Since $-f$ is an eigenvector whenever f is, every result below about $f^{+=}$ for eigenvectors also holds for $f^{-=}$.

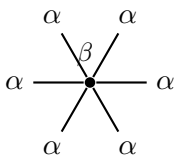
Example 19.6. Let $G = P_n$, the path $1, 2, \ldots, n$ with unit weights and unit masses. Then $v_1 = \mathbf{1}$, which has one nodal domain. By Lecture 7, for $1 \leq k \leq n$ the vector $v_k(j) = \cos(\pi(k-1)(j-\frac{1}{2})/n)$, $1 \leq j \leq n$, is an eigenvector with eigenvalue $2(1 - \cos(\pi(k-1)/n))$. These eigenvalues are distinct, so each v_k is determined up to a scalar. The entries of v_2 decrease strictly in j, from positive at $j = 1$ to negative at $j = n$, so v_2 has two nodal domains. For $n \geq 3$, the vector v_3 is positive at $j = 1$ and at $j = n$, and negative exactly for $n/4 + 1/2 < j < 3n/4 + 1/2$, an interval of length $n/2 > 1$ that contains an integer; so v_3 has three nodal domains.



Theorem 19.7 (Fiedler [Fie75])**.** *Let G be a connected graph with positive edge weights and positive vertex masses, and let $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of $L_G x = \lambda M x$. For every $k \geq 2$, every eigenvector v_k of λ_k has at most $k - 1$ weak nodal domains.*

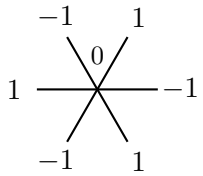
We prove Theorem 19.7 in Section 19.5. The corresponding statement for nodal domains is false, as the star shows.

Example 19.8. Let $G = S_n$ be the star with center c and $n - 1 \geq 2$ leaves, with unit weights and unit masses. Its eigenvalues are $\lambda(S_n) = (0, 1, \ldots, 1, \lambda_n)$, where λ_n belongs to the vector with value β at the center and α at every leaf, with $(n-1)\alpha = -\beta$:



Indeed, at a leaf $(Lx)_{\text{leaf}} = \alpha - \beta = n\alpha$, and at the center $(Lx)_c = (n-1)(\beta - \alpha) = n\beta$, so $\lambda_n = n$. A vector x is an eigenvector for the eigenvalue 1 if and only if $x(c) = 0$ and the leaf values add up to 0: the leaf rows of $Lx = x$ read $x(\ell) - x(c) = x(\ell)$, and then the center row reads $-\sum_\ell x(\ell) = x(c) = 0$. Every such eigenvector, for $\lambda_2, \ldots, \lambda_{n-1}$, has exactly one weak nodal domain, because $c \in x^{+=}$ is adjacent to every leaf. But c lies in neither x^+ nor x^-, so each leaf with a nonzero value is a nodal domain by itself, and the number of nodal domains of x is its number of nonzero leaves, which can be any number from 2 to $n - 1$; up to $n - 2$ of these nodal

domains can have the same sign. For example, for odd n the vector with value 0 at the center and values ± 1 at the leaves, $(n-1)/2$ of each sign, has $(n-1)/2$ positive and $(n-1)/2$ negative nodal domains:



Since $\lambda_2 = 1$, such an eigenvector of λ_2 has more than $k - 1 = 1$ nodal domains.

We return to nodal domains in Section 19.6; first we study weak nodal domains.

19.3 Generalized Laplacians

We prove Fiedler's theorem for a larger class of matrices, which need not annihilate the vector $\mathbf{1}$.

Definition 19.9. A real symmetric $n \times n$ matrix Q is a *generalized Laplacian* (GL) if

1) $Q_{ij} \le 0$ for all $i \neq j$, and

2) the graph $G(Q)$ on $\{1, \dots, n\}$ that has an edge (i, j), $i \neq j$, whenever $Q_{ij} \neq 0$ is connected.

The diagonal entries are not constrained.

The Laplacian of a connected graph G with positive weights is a GL. So is $M^{-1/2} L_G M^{-1/2}$ for every diagonal matrix M with positive diagonal entries $m(v)$, because its off-diagonal entries $-w_{uv}/\sqrt{m(u)m(v)}$ are nonpositive, and they are nonzero exactly on the edges of G.

Corollary 19.10 (Perron–Frobenius)**.** *Let Q be a GL and let λ_1 be its smallest eigenvalue. Then*

1) λ_1 has multiplicity one, and

2) every eigenvector of λ_1 is strictly positive or strictly negative.

Proof. For $n = 1$ there is nothing to prove, so let $n \ge 2$. Let $\sigma = \max_i Q_{ii}$, the least σ such that the matrix $A = \sigma I - Q$ has nonnegative entries. The matrix A is symmetric, and its graph is $G(Q)$, which is connected. By the Perron–Frobenius theorem (Theorem 5.13), $\alpha_1 = \rho(A)$ is a simple eigenvalue of A with an eigenvector $v_1 > 0$. Since A is symmetric, its eigenvalues are real, so α_1 is the largest of them.

Claim 19.11. *v_1 is an eigenvector of Q, with eigenvalue $\sigma - \alpha_1$.*

Proof of the claim.

$$\begin{aligned} (\sigma I - Q)v_1 &= \alpha_1 v_1, \\ \sigma v_1 - Q v_1 &= \alpha_1 v_1, \\ Q v_1 &= (\sigma - \alpha_1) v_1. \end{aligned}$$

The eigenvalues of Q are the numbers $\sigma - \alpha$ for the eigenvalues α of A, with the same eigenvectors and multiplicities. Since α_1 is the largest eigenvalue of A and is simple, $\lambda_1 = \sigma - \alpha_1$ is the smallest eigenvalue of Q and is simple. Every eigenvector of λ_1 is therefore a nonzero multiple of v_1, hence strictly positive or strictly negative. □

19.4 Cauchy's interlacing theorem

Theorem 19.12 (Cauchy's interlacing theorem)**.** *Let A be a real symmetric $n \times n$ matrix with $n \geq 2$, and let B be the matrix obtained from A by deleting row j and column j for some j. Let $\alpha_1 \leq \alpha_2 \leq \cdots \leq \alpha_n$ and $\beta_1 \leq \beta_2 \leq \cdots \leq \beta_{n-1}$ be the eigenvalues of A and of B. Then*

$$\alpha_1 \leq \beta_1 \leq \alpha_2 \leq \beta_2 \leq \cdots \leq \beta_{n-1} \leq \alpha_n.$$

Proof. Permuting the rows and the columns of A by the same permutation π replaces A by $P_\pi^\top A P_\pi$ for a permutation matrix P_π, which has the same eigenvalues. Hence we may assume that we delete the last row and column. For $\bar{x} \in \mathbb{R}^{n-1}$, the vector $x = (\bar{x}, 0) \in \mathbb{R}^n$ satisfies $x^\top A x = \bar{x}^\top B \bar{x}$ and $x^\top x = \bar{x}^\top \bar{x}$. By Courant–Fischer (Theorem 9.3),

$$\alpha_1 = \min_{x \neq 0} \frac{x^\top A x}{x^\top x} \leq \min_{\substack{x \neq 0 \\ x_n = 0}} \frac{x^\top A x}{x^\top x} = \min_{\bar{x} \neq 0} \frac{\bar{x}^\top B \bar{x}}{\bar{x}^\top \bar{x}} = \beta_1.$$

The same argument gives $\alpha_i \leq \beta_i$ for every $1 \leq i \leq n-1$. In increasing order, Courant–Fischer reads $\alpha_i = \min_{\dim S = i} \max_{0 \neq x \in S} x^\top A x / x^\top x$. For each subspace $S' \subseteq \mathbb{R}^{n-1}$ of dimension i, the subspace $S = \{(\bar{x}, 0) : \bar{x} \in S'\}$ of $\mathbb{R}^n$ has dimension i, and the Rayleigh quotients of A on S are those of B on S'. Hence $\alpha_i \leq \beta_i$. Applying this to $-A$ and $-B$, whose eigenvalues in increasing order are $-\alpha_n \leq \cdots \leq -\alpha_1$ and $-\beta_{n-1} \leq \cdots \leq -\beta_1$, gives $-\alpha_{n+1-i} \leq -\beta_{n-i}$ for $1 \leq i \leq n-1$, that is, $\beta_j \leq \alpha_{j+1}$ for $1 \leq j \leq n-1$. □

Corollary 19.13. *Let B be a principal submatrix of the real symmetric matrix A, obtained by deleting r rows and the same r columns, and let $\beta_1 \leq \cdots \leq \beta_{n-r}$ be its eigenvalues. Then $\alpha_i \leq \beta_i$ for $1 \leq i \leq n-r$.*

Proof. Delete the rows and columns one at a time. By Theorem 19.12, no deletion decreases the i-th smallest eigenvalue, for any i that remains in range. □

19.5 Fiedler's theorem for generalized Laplacians

Theorem 19.14. *Let L be an $n \times n$ generalized Laplacian with eigenvalues $\lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$, and let $G = G(L)$. For every $k \geq 2$, every eigenvector z_k of λ_k has at most $k-1$ weak nodal domains in G.*

Proof. Let $L z_k = \lambda_k z_k$ with $z_k \neq 0$, and let

$$W_k = W_{z_k}^+ = \{v \in V \mid z_k(v) \geq 0\}.$$

By Corollary 19.10, λ_1 is simple with an eigenvector $z_1 > 0$. Since L is symmetric and $\lambda_k \neq \lambda_1$, we have $z_k \perp z_1$, so z_k has both positive and negative entries; in particular $W_k \neq \emptyset$ and $W_k \neq V$.

Suppose that $G(W_k)$ has t connected components $W_1, \ldots, W_t$. We order the vertices by component, followed by the vertices of $V \setminus W_k$, and write

$$L = \begin{pmatrix} B_1 & 0 & \cdots & 0 & C_1 \\ 0 & B_2 & \cdots & 0 & C_2 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & B_t & C_t \\ C_1^\top & C_2^\top & \cdots & C_t^\top & D \end{pmatrix}, \qquad z_k = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_t \\ y \end{pmatrix}, \qquad L z_k = \lambda_k z_k,$$

where x_i and B_i belong to the i-th weak nodal domain W_i. The blocks between W_i and W_j for $i \neq j$ are zero, because no edge of G joins two different components of $G(W_k)$. We use four simple facts.

1) Each B_i is a generalized Laplacian: it is symmetric, its off-diagonal entries are nonpositive, and its graph is $G(W_i)$, which is connected.

2) Each C_i satisfies $C_i \leq 0$ entrywise and $C_i \neq 0$. Indeed, G is connected and $W_i \neq V$, so some edge joins W_i to a vertex outside W_i; this vertex is not in another W_j, so it lies in $V \setminus W_k$.

3) $x_i \geq 0$ and $y < 0$ entrywise.

4) The rows of D are indexed by $\{v \in V \mid z_k(v) < 0\} = V \setminus W_k$, the entries of y.

Claim 19.15. *$\lambda_{\min}(B_i) < \lambda_k$ for every i.*

Proof. The rows of $Lz_k = \lambda_k z_k$ that belong to W_i read

$$B_i x_i + C_i y = \lambda_k x_i.$$

Since $C_i \leq 0$, $C_i \neq 0$ and $y < 0$, the vector $C_i y$ satisfies $C_i y \geq 0$ and $C_i y \neq 0$: a row of C_i with a negative entry gives a positive entry of $C_i y$. In particular $x_i \neq 0$, since $x_i = 0$ would give $C_i y = \lambda_k x_i - B_i x_i = 0$. Hence

$$B_i x_i = \lambda_k x_i - C_i y \leq \lambda_k x_i$$

entrywise. Since $x_i \geq 0$, multiplying entrywise by x_i and adding gives $x_i^\top B_i x_i \leq \lambda_k x_i^\top x_i$. By Courant–Fischer,

$$\lambda_1(B_i) = \min_{x \neq 0} \frac{x^\top B_i x}{x^\top x} \leq \frac{x_i^\top B_i x_i}{x_i^\top x_i} \leq \lambda_k.$$

To show that the inequality is strict, we consider two cases.

Case 1: x_i has a zero entry. By Corollary 19.10 applied to the generalized Laplacian B_i, every eigenvector of $\lambda_1(B_i)$ is strictly positive or strictly negative, so x_i is not one. The Rayleigh quotient of a symmetric matrix attains its minimum only at eigenvectors of the smallest eigenvalue, so $\lambda_1(B_i) < x_i^\top B_i x_i / x_i^\top x_i \leq \lambda_k$.

Case 2: x_i is strictly positive. Then $x_i^\top C_i y > 0$, and

$$x_i^\top B_i x_i < x_i^\top B_i x_i + x_i^\top C_i y = \lambda_k x_i^\top x_i.$$

Thus $\lambda_1(B_i) \leq x_i^\top B_i x_i / x_i^\top x_i < \lambda_k$. □

By Claim 19.15, for each i there exist $\mu_i < \lambda_k$ and $\bar{w}_i \neq 0$ with $B_i \bar{w}_i = \mu_i \bar{w}_i$. Consider the principal submatrix of L on W_k and the vectors obtained by padding $\bar{w}_i$ with zeros:

$$\begin{bmatrix} B_1 & & 0 \\ & \ddots & \\ 0 & & B_t \end{bmatrix} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \bar{w}_i \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mu_i \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \bar{w}_i \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

These t eigenvectors have disjoint supports, so they are linearly independent, and the submatrix has at least t eigenvalues smaller than λ_k, counted with multiplicity. By Corollary 19.13, the t-th smallest eigenvalue of L is at most the t-th smallest eigenvalue of the submatrix, so it is smaller than λ_k. Since λ_k is the k-th smallest eigenvalue of L, this forces $t \le k-1$. □

Proof of Theorem 19.7. The matrix $L' = M^{-1/2} L_G M^{-1/2}$ is a generalized Laplacian with $G(L') = G$, and it has the same eigenvalues as the problem $L_G x = \lambda M x$. A vector x is an eigenvector of that problem for λ_k if and only if $M^{1/2}x$ is an eigenvector of L' for λ_k. Since $M^{1/2}$ is diagonal with positive entries, x and $M^{1/2}x$ have the same sign at every vertex, hence the same weak nodal domains. Now apply Theorem 19.14. □

19.6 Back to nodal domains

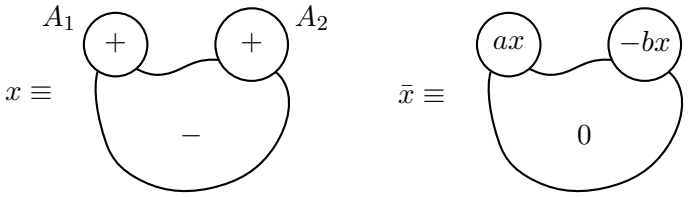
Definition 19.16. The *support* of $f : V \to \mathbb{R}$ is $\operatorname{support}(f) = \{v \in V \mid f(v) \neq 0\}$. Its positive and negative parts are the sets f^+ and f^- of Definition 19.1, so that $\operatorname{support}(f) = f^+ \cup f^-$.

Definition 19.17. Let A be a real symmetric matrix and x an eigenvector of A for the eigenvalue λ. Then x has *minimal support* if there is no eigenvector y of A for λ with $\operatorname{support}(y) \subsetneq \operatorname{support}(x)$.

Theorem 19.18. *Let Q be a generalized Laplacian with graph $G = G(Q)$ and eigenvalues $\lambda_1 < \lambda_2 \le \cdots \le \lambda_n$, and let x be an eigenvector of λ_2 with minimal support. Then $G(x^+)$ and $G(x^-)$ are connected.*

Since x takes both signs (as in the proof of Theorem 19.14), such an x has exactly two nodal domains.

The idea of the proof is the following. Suppose that x is an eigenvector with two positive nodal domains, as in the first picture below. Consider the vector $\bar{x}$ that agrees with x on one of them, with $-x$ on the other, and vanishes elsewhere, and adjust the two parts so that $\bar{x}$ is orthogonal to the first eigenvector ($\bar{x} \perp \mathbf{1}$ for a Laplacian).



The following claim makes this precise; it does not use minimality.

Claim 19.19. *Let Q and λ_2 be as in Theorem 19.18, let $z_1 > 0$ be an eigenvector of λ_1, and let x be an eigenvector of λ_2 such that $G(x^+)$ is not connected. Let A_1 be the vertex set of one component of $G(x^+)$, let $A_2 = x^+ \setminus A_1$, and for $j = 1, 2$ let $x^{(j)}$ agree with x on A_j and vanish elsewhere. Set $a = z_1^\top x^{(2)}$, $b = z_1^\top x^{(1)}$ and $\bar{x} = a x^{(1)} - b x^{(2)}$. Then $\bar{x} \neq 0$, $\bar{x} \perp z_1$,*

$$\frac{\bar{x}^\top Q \bar{x}}{\bar{x}^\top \bar{x}} \le \frac{x^\top Q x}{x^\top x} \quad \text{and} \quad \operatorname{support}(\bar{x}) \subsetneq \operatorname{support}(x).$$

Proof. The sets A_1 and A_2 are nonempty, and $x^{(1)}, x^{(2)} \ge 0$ are nonzero with disjoint supports. Since $z_1 > 0$, we have $a, b > 0$, so $\bar{x} \neq 0$, and $z_1^\top \bar{x} = ab - ba = 0$. The support of $\bar{x}$ is $A_1 \cup A_2 = x^+$, which is a proper subset of $\operatorname{support}(x)$ because $x^- \neq \emptyset$ (as in the proof of Theorem 19.14).

No edge of G joins A_1 to A_2, so $Q_{uv} = 0$ for $u \in A_1$ and $v \in A_2$, and

$$\bar{x}^\top Q \bar{x} = a^2\, x^{(1)\top} Q x^{(1)} + b^2\, x^{(2)\top} Q x^{(2)}.$$

Fix $j \in \{1, 2\}$ and $u \in A_j$. Then

$$\lambda_2 x_u = (Qx)_u = (Qx^{(j)})_u + \sum_{v \notin A_j} Q_{uv} x_v.$$

If $v \notin A_j$ and $Q_{uv} \neq 0$, then v is a neighbor of u outside A_j. Since A_j is a union of components of $G(x^+)$, the vertex v is not in x^+, so $x_v \leq 0$ and $Q_{uv} x_v \geq 0$. Hence $(Qx^{(j)})_u \leq \lambda_2 x_u$. Multiplying by $x_u > 0$ and adding over $u \in A_j$ gives $x^{(j)\top} Q x^{(j)} \leq \lambda_2\, x^{(j)\top} x^{(j)}$. Therefore

$$\bar{x}^\top Q \bar{x} \leq \lambda_2 \big(a^2\, x^{(1)\top} x^{(1)} + b^2\, x^{(2)\top} x^{(2)}\big) = \lambda_2\, \bar{x}^\top \bar{x},$$

and $\lambda_2 = x^\top Q x / x^\top x$ because $Qx = \lambda_2 x$. □

Proof of Theorem 19.18. Suppose that $G(x^+)$ is not connected, and let $\bar{x}$ be as in Claim 19.19. Let $u_1 = z_1/\|z_1\|, u_2, \ldots, u_n$ be orthonormal eigenvectors of Q with $Qu_i = \lambda_i u_i$. Since $\bar{x} \perp z_1$, we can write $\bar{x} = \sum_{i \geq 2} c_i u_i$, and

$$\lambda_2 \geq \frac{\bar{x}^\top Q \bar{x}}{\bar{x}^\top \bar{x}} = \frac{\sum_{i \geq 2} c_i^2 \lambda_i}{\sum_{i \geq 2} c_i^2} \geq \lambda_2.$$

So equality holds throughout, which forces $c_i = 0$ whenever $\lambda_i > \lambda_2$. Hence $\bar{x}$ is an eigenvector of λ_2 with $\mathrm{support}(\bar{x}) \subsetneq \mathrm{support}(x)$, which contradicts the minimality of the support of x. Thus $G(x^+)$ is connected. The vector $-x$ is also an eigenvector of λ_2 with minimal support, and $(-x)^+ = x^-$, so $G(x^-)$ is connected as well. □

20 Eigenvalues and Eigenvectors by Iterative Methods (Power and Inverse Iteration)

Lecture of March 20, 2020. Based on Gary L. Miller's handwritten notes.

Spectral methods often need an eigenvector rather than the solution of a linear system, for example the Fiedler vector, an eigenvector of λ_2 of a Laplacian. This lecture analyzes the two simplest iterative methods for computing one eigenvalue and eigenvector. Power iteration repeatedly multiplies a random unit vector by A; its error decays like $(\lambda_2/\lambda_1)^{2k}$, which is too slow for the Fiedler vector of a path. Inverse iteration solves a linear system in each step instead, which is worthwhile when a fast solver is available (Lectures 17 and 18); it approximates the smallest nonzero eigenvalue μ_2 within a factor $1+\varepsilon$ after $O(\varepsilon^{-1}\log(n/\varepsilon))$ iterations, with probability greater than $1/2$. Both analyses use a fact about the first coordinate of a random unit vector. We end by showing that computing eigenvalues is at least as hard as finding the roots of polynomials.

20.1 Iterative methods for eigenvalues

In this subsection and the next, A is a real symmetric positive semidefinite (SPSD) $n \times n$ matrix. Suppose that we want its largest eigenvalue and a corresponding eigenvector. The simplest method is powering.

Algorithm 20.1 (Power iteration)**.** Input: an SPSD matrix A.
Initialization: choose $v^{(0)}$ uniformly at random on the unit sphere $S^{n-1} = \{v \in \mathbb{R}^n : \|v\| = 1\}$.
For $k = 1, 2, \ldots$:

1) $w = Av^{(k-1)}$ (a sparse matrix-vector product);

2) $v^{(k)} = w/\|w\|$ (normalization);

3) $\lambda^{(k)} = v^{(k)\top} A v^{(k)}$ (the Rayleigh quotient of $v^{(k)}$).

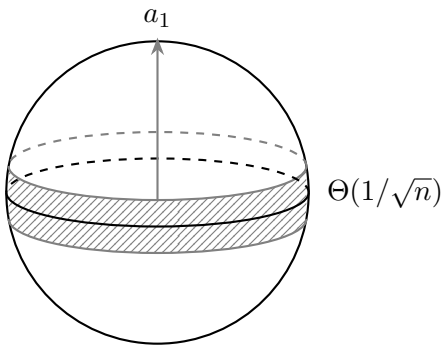
20.2 Analysis

20.2.1 Initialization

Power iteration amplifies the component of $v^{(0)}$ along the top eigenvector, so we first ask how large this component is for a random start. By rotation invariance, it is distributed like the first coordinate a_1 of a uniformly random point $v = (a_1, \ldots, a_n)$ of S^{n-1}. Since $a_1^2 + \cdots + a_n^2 = 1$ and the a_i^2 have the same distribution by symmetry, $\mathbb{E}[a_1^2] = 1/n$. So a_1 is typically of order $1/\sqrt{n}$, and the following fact says that it is not much smaller with probability greater than one half.

Fact 20.2. *Let $n \geq 1$ and let $(a_1, \ldots, a_n)$ be uniformly distributed on S^{n-1}. Then* $\Pr\left[|a_1| \geq \frac{2}{3\sqrt{n}}\right] > \frac{1}{2}$.

We state this fact without proof. The picture behind it is that the band of the sphere where $|a_1|$ is below a constant times $1/\sqrt{n}$ has width $\Theta(1/\sqrt{n})$, and for the constant $2/3$ it carries less than half of the area.



As $n \to \infty$, the variable $\sqrt{n}\, a_1$ converges in distribution to a standard normal variable, so the probability in Fact 20.2 tends to $2(1 - \Phi(2/3)) \approx 0.505$, where Φ is the standard normal distribution function. The fact therefore holds with little room to spare.

20.2.2 The iterative step

Suppose that the eigenvalues of A are $\lambda_1 > \lambda_2 \geq \cdots \geq \lambda_n > 0$, with orthonormal eigenvectors $q_1, \ldots, q_n$, and write $v^{(0)} = a_1 q_1 + \cdots + a_n q_n$. By rotation invariance, $(a_1, \ldots, a_n)$ is uniformly distributed on S^{n-1}. Normalization only rescales $v^{(k)}$, and the Rayleigh quotient does not change under rescaling. So we may consider power iteration without normalization, $v^{(k)} = A^k v^{(0)}$, and

$$\lambda^{(k)} = \frac{v^{(k)\top} A v^{(k)}}{v^{(k)\top} v^{(k)}}.$$

By orthonormality,

$$\begin{aligned} v^{(k)} &= a_1 \lambda_1^k q_1 + \cdots + a_n \lambda_n^k q_n, \\ v^{(k)\top} v^{(k)} &= a_1^2 \lambda_1^{2k} + \cdots + a_n^2 \lambda_n^{2k}, \\ v^{(k)\top} A v^{(k)} &= a_1^2 \lambda_1^{2k+1} + \cdots + a_n^2 \lambda_n^{2k+1}, \end{aligned}$$

and therefore

$$\lambda^{(k)} = \frac{\sum_i a_i^2 \lambda_i^{2k+1}}{\sum_i a_i^2 \lambda_i^{2k}}. \tag{20.1}$$

Let $\varepsilon^{(k)} = \lambda_1 - \lambda^{(k)}$.

Claim 20.3. *If $a_1 \neq 0$, then for every $k \geq 0$,*

$$0 \leq \varepsilon^{(k)} \leq \frac{\lambda_1 - \lambda_n}{a_1^2} \left(\frac{\lambda_2}{\lambda_1}\right)^{2k}.$$

In particular, $\varepsilon^{(k)} = O\big((\lambda_2/\lambda_1)^{2k}\big)$ as $k \to \infty$.

Proof. By (20.1), $\lambda^{(k)}$ is a weighted average of the λ_i, so $\lambda^{(k)} \le \lambda_1$ and $\varepsilon^{(k)} \ge 0$. Next,

$$\begin{aligned}
\varepsilon^{(k)} = \lambda_1 - \lambda^{(k)} &= \frac{\lambda_1\left(\sum_i a_i^2\lambda_i^{2k}\right) - \sum_i a_i^2\lambda_i^{2k+1}}{\sum_i a_i^2\lambda_i^{2k}} \\
&\overset{(a)}{\le} \frac{\lambda_1\left(\sum_i a_i^2\lambda_i^{2k}\right) - a_1^2\lambda_1^{2k+1} - \left(\sum_{i>1} a_i^2\lambda_i^{2k}\right)\lambda_n}{\sum_i a_i^2\lambda_i^{2k}} \\
&= \frac{\left(\sum_{i>1} a_i^2\lambda_i^{2k}\right)(\lambda_1 - \lambda_n)}{a_1^2\lambda_1^{2k} + \sum_{i>1} a_i^2\lambda_i^{2k}} \\
&\overset{(b)}{\le} \frac{\left(\sum_{i=1}^n a_i^2\lambda_2^{2k}\right)(\lambda_1 - \lambda_n)}{a_1^2\lambda_1^{2k} + \sum_{i=1}^n a_i^2\lambda_2^{2k}} = \frac{(\lambda_1 - \lambda_n)\lambda_2^{2k}}{a_1^2\lambda_1^{2k} + \lambda_2^{2k}} \le \frac{\lambda_1 - \lambda_n}{a_1^2}\left(\frac{\lambda_2}{\lambda_1}\right)^{2k}.
\end{aligned}$$

Step (a) uses $\lambda_i^{2k+1} \ge \lambda_i^{2k}\lambda_n$ for $i > 1$. Step (b) holds because $y \mapsto (\lambda_1 - \lambda_n)\, y/(a_1^2\lambda_1^{2k} + y)$ is nondecreasing for $y \ge 0$, and $\sum_{i>1} a_i^2\lambda_i^{2k} \le \lambda_2^{2k}\sum_{i>1} a_i^2 \le \lambda_2^{2k}\sum_{i=1}^n a_i^2 = \lambda_2^{2k}$. □

This rate is not good enough for computing the Fiedler vector. Let $L = L(P_n)$ with $n \ge 3$. By Lecture 7, the eigenvalues of L are $2(1 - \cos(\pi j/n))$ for $j = 0, 1, \ldots, n-1$. For small $j \ge 1$ this is approximately $\pi^2/(n/j)^2$, so the spectrum begins $0, \pi^2/n^2, \pi^2/(n/2)^2, \pi^2/(n/3)^2, \ldots$, and the largest eigenvalue is $2(1 + \cos(\pi/n)) < 4$. We want the eigenvector q_2 of $\lambda_2 \approx \pi^2/n^2$, so we need q_2 to be the top eigenvector. Try $M = 4I - L$. It is positive semidefinite and maps $\mathbf{1}^\perp$ to itself; starting with $v^{(0)} \perp \mathbf{1}$ removes its eigenvalue 4 on $\mathbf{1}$. On $\mathbf{1}^\perp$, the largest eigenvalue of M is $4 - \lambda_2 \approx 4 - \pi^2/n^2$, with eigenvector q_2, and the next one is $4 - \lambda_3 \approx 4 - 4\pi^2/n^2$. But

$$\frac{4 - \lambda_3}{4 - \lambda_2} \approx \frac{4n^2 - 4\pi^2}{4n^2 - \pi^2};$$

precisely, $(4-\lambda_3)/(4-\lambda_2) \ge 1 - \lambda_3/(4-\lambda_2) \ge 1 - 2\pi^2/n^2$, since $\lambda_3 = 2(1-\cos(2\pi/n)) \le 4\pi^2/n^2$ by $1 - \cos y \le y^2/2$, and $4 - \lambda_2 \ge 2$. So the bound of Claim 20.3 shrinks by a constant factor only every $\Omega(n^2)$ iterations: we need about n^2 iterations per bit.

20.3 Inverse iteration

If we have a fast solver for systems with matrix A, we can use it to find the smallest nonzero eigenvalue instead. In this subsection, A is an SPSD $n \times n$ matrix, $n \ge 2$, whose null space is exactly $\operatorname{span}\{\mathbf{1}\}$, as for the Laplacian of a connected graph. Its eigenvalues are $0 = \mu_1 < \mu_2 \le \cdots \le \mu_n$, with orthonormal eigenvectors $q_1 = \mathbf{1}/\sqrt{n}, q_2, \ldots, q_n$.

Algorithm 20.4 (Inverse iteration)**.** Input: A as above and $\varepsilon \in (0, 1]$.

Initialization: choose $v^{(0)}$ uniformly at random among the unit vectors orthogonal to $\mathbf{1}$.

For $i = 1$ to k, where $k = \left\lceil \frac{c}{\varepsilon} \ln\left(\frac{cn}{\varepsilon}\right) \right\rceil$ for a suitable constant c ($c = 9/2$ suffices, by Theorem 20.5):

($*$) solve $Av^{(i)} = v^{(i-1)}$, taking the solution with $v^{(i)} \perp \mathbf{1}$.

Return $v^{(k)}$ and $\lambda^{(k)} = \dfrac{v^{(k)\top} A v^{(k)}}{v^{(k)\top} v^{(k)}}$.

The system in step ($*$) is solvable because $v^{(i-1)}$ is orthogonal to $\mathbf{1}$, hence lies in the range of A. If a solution is not orthogonal to $\mathbf{1}$, we project out its component along $\mathbf{1}$; the result is $A^\dagger v^{(i-1)}$. We may also rescale $v^{(i)}$ when it becomes too large, since rescaling does not change $\lambda^{(k)}$.

Theorem 20.5. *Let A be as above, let $0 < \varepsilon \le 1$, and let k be an integer with $k \ge \frac{2}{\varepsilon}\ln(\frac{9n}{2\varepsilon})$. With probability greater than $1/2$ over the choice of $v^{(0)}$, inverse iteration returns*

$$\lambda^{(k)} \le (1+\varepsilon)\mu_2.$$

Proof. Write

$$r = v^{(0)} = \alpha_2 q_2 + \cdots + \alpha_n q_n;$$

there is no q_1 term because $v^{(0)} \perp \mathbf{1}$. Up to a scalar factor, $v^{(k)} = (A^\dagger)^k v^{(0)}$, so

$$v^{(k)} = \alpha_2 \mu_2^{-k} q_2 + \cdots + \alpha_n \mu_n^{-k} q_n,$$
$$v^{(k)\top} v^{(k)} = \sum_{i\ge 2} \frac{\alpha_i^2}{\mu_i^{2k}},$$

and

$$\lambda^{(k)} = \frac{v^{(k)\top} A v^{(k)}}{v^{(k)\top} v^{(k)}} = \frac{\sum_{i\ge 2} \alpha_i^2/\mu_i^{2k-1}}{\sum_{i\ge 2} \alpha_i^2/\mu_i^{2k}} =: (**). \tag{20.2}$$

Assume that $\alpha_2 \neq 0$; this holds in the event considered below. We pick

$$j = \max\Big\{ i : \mu_i \le \Big(1+\frac{\varepsilon}{2}\Big)\mu_2 \Big\},$$

so that $\mu_i \le (1+\varepsilon/2)\mu_2$ for $2 \le i \le j$ and $\mu_i > (1+\varepsilon/2)\mu_2$ for $i > j$. Split the numerator of $(**)$ as $N_1 + N_2$ and the denominator as $D_1 + D_2$, where N_1 and D_1 are the sums over $2 \le i \le j$, and N_2 and D_2 the sums over $i > j$. Then $(N_1+N_2)/(D_1+D_2) \le N_1/D_1 + N_2/(D_1+D_2)$, and $D_1 + D_2 \ge \alpha_2^2/\mu_2^{2k}$. Hence

$$\begin{aligned}
(**) &\le \frac{\sum_{2\le i\le j} \alpha_i^2/\mu_i^{2k-1}}{\sum_{2\le i\le j} \alpha_i^2/\mu_i^{2k}} + \frac{\sum_{i>j} \alpha_i^2/\mu_i^{2k-1}}{\sum_{i\ge 2} \alpha_i^2/\mu_i^{2k}} \\
&\le \mu_j + \frac{\sum_{i>j} \alpha_i^2/\mu_i^{2k-1}}{\alpha_2^2/\mu_2^{2k}} \\
&= \mu_j + \Big(\frac{\mu_2}{\alpha_2^2}\Big) \sum_{i>j} \alpha_i^2 \Big(\frac{\mu_2}{\mu_i}\Big)^{2k-1} \\
&\le \mu_j + \frac{\mu_2}{\alpha_2^2}\Big(\frac{\mu_2}{\mu_{j+1}}\Big)^{2k-1}. \qquad (***)
\end{aligned}$$

The first fraction is at most μ_j because it is a weighted average of the μ_i with $2 \le i \le j$: each term of its numerator is μ_i times the corresponding term of its denominator. The last step uses $\mu_i \ge \mu_{j+1}$ for $i > j$ and $\sum_i \alpha_i^2 = 1$. If $j = n$, the sums over $i > j$ are empty and the second terms are 0.

Now $\mu_j \le (1+\varepsilon/2)\mu_2$ and $\mu_2/\mu_{j+1} < 1/(1+\varepsilon/2)$, so

$$(***) \le \Big(1+\frac{\varepsilon}{2}\Big)\mu_2 + \frac{\mu_2}{\alpha_2^2}\Big(\frac{1}{1+\varepsilon/2}\Big)^{2k-1}. \qquad (\ddagger)$$

The coordinates $(\alpha_2, \ldots, \alpha_n)$ of $v^{(0)}$ in the orthonormal basis $q_2, \ldots, q_n$ of $\mathbf{1}^\perp$ are uniformly distributed on the unit sphere of $\mathbb{R}^{n-1}$. By Fact 20.2 in dimension $n-1$, with probability greater than $1/2$ we have

$$|\alpha_2| \ge \frac{2}{3\sqrt{n-1}} \ge \frac{2}{3\sqrt{n}}, \quad \text{that is,} \quad \alpha_2^2 \ge \frac{4}{9n}.$$

In this event, suppose that k satisfies

$$\frac{1}{\alpha_2^2}\Big(\frac{1}{1+\varepsilon/2}\Big)^{2k-1} \le \Big(\frac{4}{9n}\Big)^{-1}\Big(\frac{1}{1+\varepsilon/2}\Big)^{2k-1} \le \frac{\varepsilon}{2},$$

that is,

$$\Big(\frac{1}{1+\varepsilon/2}\Big)^{2k-1} \le \frac{2\varepsilon}{9n}, \quad \text{or equivalently} \quad \boxed{\Big(1+\frac{\varepsilon}{2}\Big)^{2k-1} \ge \frac{9n}{2\varepsilon}}. \tag{20.3}$$

Then

$$(\ddagger) \le \Big(1+\frac{\varepsilon}{2}\Big)\mu_2 + \frac{\varepsilon}{2}\mu_2 = (1+\varepsilon)\mu_2,$$

and thus $\lambda^{(k)} \le (1+\varepsilon)\mu_2$.

To finish the proof, we show that every integer $k \ge (2/\varepsilon)\ln(9n/(2\varepsilon))$ satisfies the boxed inequality (20.3). Set $\tau = 2k-1$. Since $0 < \varepsilon \le 1$, we have $\ln(1+\varepsilon/2) \ge (\varepsilon/2)/(1+\varepsilon/2) \ge \varepsilon/3$, that is, $(1+\varepsilon/2)^{2/\varepsilon} \ge e^{2/3}$. Hence

$$\Big(1+\frac{\varepsilon}{2}\Big)^{\tau} = \Big(\Big(1+\frac{\varepsilon}{2}\Big)^{2/\varepsilon}\Big)^{\varepsilon\tau/2} \ge e^{\varepsilon\tau/3},$$

and the boxed inequality holds as soon as

$$\frac{\varepsilon}{3}\tau \ge \ln\Big(\frac{9n}{2\varepsilon}\Big), \quad \text{that is,} \quad \tau \ge \frac{3}{\varepsilon}\ln\Big(\frac{9n}{2\varepsilon}\Big), \quad \text{or} \quad k \ge \frac{1}{2} + \frac{3}{2\varepsilon}\ln\Big(\frac{9n}{2\varepsilon}\Big).$$

Since $\ln(9n/(2\varepsilon)) \ge \ln(9/2) > 1 \ge \varepsilon$, this follows from $k \ge (2/\varepsilon)\ln(9n/(2\varepsilon))$. □

20.4 Finding all the eigenvalues and eigenvectors

Two important methods compute all the eigenvalues and eigenvectors of a matrix: the Arnoldi iteration, for general matrices, and the Lanczos iteration, for symmetric matrices.

Claim 20.6. *Computing eigenvalues is as hard as finding roots of polynomials. More precisely, for every polynomial $P(z) = z^m + a_{m-1}z^{m-1} + \cdots + a_0$ with $m \ge 1$, there is an $m \times m$ matrix C_p whose entries are 0, 1 and $-a_0, \ldots, -a_{m-1}$, such that the eigenvalues of C_p are exactly the roots of P, with multiplicity.*

Proof. Let C_p be the *companion matrix* of P,

$$C_p = \begin{pmatrix} 0 & \cdots & & 0 & -a_0 \\ 1 & 0 & \cdots & 0 & -a_1 \\ & \ddots & \ddots & \vdots & \vdots \\ 0 & & \ddots & 0 & -a_{m-2} \\ 0 & & & 1 & -a_{m-1} \end{pmatrix}.$$

We have $\det(zI - C_p) = P(z)$, so the characteristic polynomial of C_p is P. To verify this, expand the determinant along the last column. The matrix $zI - C_p$ has z on the diagonal, except for the entry $z + a_{m-1}$ in the corner, -1 on the subdiagonal, and last column $(a_0, \ldots, a_{m-2}, z + a_{m-1})^\top$. Deleting row i and the last column leaves a block lower triangular matrix whose diagonal blocks are an $(i-1) \times (i-1)$ lower bidiagonal matrix with z on the diagonal and an $(m-i) \times (m-i)$ upper bidiagonal matrix with -1 on the diagonal; its determinant is $z^{i-1}(-1)^{m-i}$. With the cofactor sign $(-1)^{i+m}$, the i-th term of the expansion is $a_{i-1}z^{i-1}$ for $i < m$ and $(z + a_{m-1})z^{m-1}$ for $i = m$, and these terms add up to $P(z)$. Hence an algorithm that computes the eigenvalues of every matrix also computes the roots of every polynomial. □

21 Graph Maximum Cut via Spectral Methods (Trevisan)

Lecture of March 23, 2020. Based on Gary L. Miller's handwritten notes.

The maximum cut problem asks for a partition of the vertices of a graph that cuts as many edges as possible. A uniformly random partition cuts half of the edges in expectation, so a $\frac{1}{2}$-approximation is easy, and the question is whether a spectral method can do better. Cheeger's inequality (Lecture 12) relates the second-smallest eigenvalue of the normalized Laplacian to sparse cuts. This lecture proves Trevisan's analogue for the largest eigenvalue, which relates it to cuts that cut almost every edge. The proof rounds an eigenvector to a vector with entries in $\{0, \pm 1\}$ and is constructive, and it leads to a recursive algorithm that cuts, in expectation, at least 0.529 times the maximum number of edges.

21.1 Spectral maximum cut

Throughout this lecture, $G = (V, E)$ is an undirected graph with $V = \{1, \dots, n\}$, $n \geq 2$, edge weights $w_{ij} > 0$, and no isolated vertices. Its weighted degrees d_i are therefore positive, so the degree matrix D is positive definite. As in Lecture 12, A is the adjacency matrix and $L = D - A$ is the Laplacian of G. The *normalized Laplacian* problem is the generalized eigenvalue problem $Lx = \lambda Dx$. Its eigenvalues are real and nonnegative, and we order them as

$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n;$$

if G is connected, then $\lambda_1 < \lambda_2$. They are the eigenvalues of the matrix $\tilde{L}_G = D^{-1/2} L D^{-1/2}$, because $Lx = \lambda Dx$ holds if and only if $\tilde{L}_G(D^{1/2}x) = \lambda D^{1/2}x$.

Theorem 21.1 (Cheeger's inequality)**.** *If G is connected, then its conductance $\Phi(G)$, as defined in Lecture 12, satisfies*

$$\tfrac{1}{2}\lambda_2 \leq \Phi(G) \leq \sqrt{2\lambda_2}.$$

This is proved in Lecture 12. Our goal is a theorem of the same kind for the maximum cut problem.

Definition 21.2 (Max-Cut problem)**.** For $x \in \{\pm 1\}^n$, the *cut* of x is the set of edges $(i, j) \in E$ with $x_i \neq x_j$, that is, the edges between $\{i : x_i = 1\}$ and $\{i : x_i = -1\}$. The *Max-Cut problem* is to find $x \in \{\pm 1\}^n$ that maximizes

$$\sum_{(i,j)\in E} |x_i - x_j|\, w_{ij} = \frac{1}{2} \sum_{(i,j)\in E} (x_i - x_j)^2\, w_{ij}.$$

We write $\mathrm{MaxCut}(G)$ for the maximum total weight of the edges of a cut.

For $x \in \{\pm 1\}^n$, a cut edge has $|x_i - x_j| = 2$ and $(x_i - x_j)^2 = 4$, and every other edge contributes 0 to both sums. So the two sums are equal, and each counts every cut edge twice; their common maximum is $2\,\mathrm{MaxCut}(G)$. An algorithm is an *α-approximation* if the cut it returns has weight at least $\alpha\,\mathrm{MaxCut}(G)$; for a randomized algorithm, we ask this of the expected weight.

If x is chosen uniformly at random from $\{\pm 1\}^n$, then $\Pr[x_i \neq x_j] = \frac{1}{2}$ for every edge (i, j), so the expected weight of the cut of x is half of the total edge weight. In particular, with unit weights,

$$\mathrm{MaxCut}(G) \geq m/2,$$

where $m = |E|$. Hence a 0.5-approximation is easy: a uniformly random partition is one. Our goal is a $(0.5 + \varepsilon)$-approximation for some constant $\varepsilon > 0$.

Remark 21.3. Goemans and Williamson [GW95] obtain a 0.878-approximation for Max-Cut using semidefinite programming (SDP).

From now on we assume unit edge weights, $w_{ij} = 1$, so that d_i is the number of edges at i. The *sum Laplacian* of G is

$$P = D + A = 2D - L.$$

We write $\beta_1 \le \beta_2 \le \cdots \le \beta_n$ for the eigenvalues of the generalized problem $Px = \beta Dx$.

Theorem 21.4. *$\lambda_n \le 2$. If G is connected, then $\lambda_n = 2$ if and only if G is bipartite.*

Proof. Let $\bar{B}$ be the $m \times n$ matrix with one row for each edge: the row of the edge (i, j) has entries 1 in columns i and j and 0 elsewhere. Then $(\bar{B}^\top \bar{B})_{ii} = d_i$, and for $i \ne j$ the entry $(\bar{B}^\top \bar{B})_{ij}$ is the number of edges between i and j. Hence $P = \bar{B}^\top \bar{B}$, so P is symmetric positive semidefinite (SPSD), and since the entry of $\bar{B}x$ at the edge (i, j) is $x_i + x_j$,

$$x^\top P x = \|\bar{B}x\|^2 = \sum_{(i,j)\in E} (x_i + x_j)^2. \tag{21.1}$$

If $Px = \beta Dx$ with $x \ne 0$, then $\beta = x^\top Px / x^\top Dx \ge 0$, so $0 \le \beta_1 \le \cdots \le \beta_n$. Moreover,

$$(2D - L)x = \beta Dx \quad \text{if and only if} \quad Lx = (2 - \beta)Dx,$$

so the eigenvalues of the two problems correspond through $\beta = 2 - \lambda$, with the same eigenvectors, and the correspondence reverses the order. In particular $\beta_1 = 2 - \lambda_n$, that is, $\beta_1 + \lambda_n = 2$, and the eigenvectors for β_1 are those for λ_n. Since $\beta_1 \ge 0$, we get $\lambda_n \le 2$.

Now let G be connected. Since $\beta_1 = 2 - \lambda_n$, we have $\lambda_n = 2$ if and only if $\beta_1 = 0$. Since P is SPSD, this holds if and only if $x^\top Px = 0$ for some $x \ne 0$, that is, by (21.1), $x_i = -x_j$ on every edge (i, j). Such an x has $|x_i| = |x_j|$ on every edge, so, since G is connected, all $|x_i|$ are equal to some $c > 0$, and the signs of x alternate along every edge. Thus the sets $\{x_i > 0\}$ and $\{x_i < 0\}$ form a bipartition of G. Conversely, if (V_1, V_2) is a bipartition of G, the vector that is 1 on V_1 and -1 on V_2 satisfies $x_i = -x_j$ on every edge. □

A first idea for a spectral Max-Cut algorithm is the following:

1) compute an eigenvector x for β_1, which is also an eigenvector for λ_n;

2) sort the vertices by their values in x;

3) return the best threshold cut, that is, among the cuts between $\{i : x_i \le \tau\}$ and $\{i : x_i > \tau\}$ over all thresholds τ, one that cuts the most edges.

The following example shows that this can fail badly.

Example 21.5. Let $G = K_n$ with $n \ge 4$ even. Here $D = (n-1)I$ and $L = nI - J$, where J is the all-ones matrix.

1) $\mathrm{MaxCut}(K_n) = (n/2)^2$: a cut between sets of sizes k and $n - k$ has $k(n - k)$ edges, and this is largest at $k = n/2$.

2) $\lambda(L_G) = \{0, n, \ldots, n\}$: we have $L\mathbf{1} = 0$, and $Lx = nx$ for every $x \perp \mathbf{1}$.

3) $\lambda(\tilde{L}_G) = \{0, \frac{n}{n-1}, \ldots, \frac{n}{n-1}\}$, because $D = (n-1)I$.

Hence $\lambda_n = \frac{n}{n-1}$ and $\beta_1 = 2 - \frac{n}{n-1}$, and every nonzero $x \perp \mathbf{1}$ is an eigenvector for β_1. In particular, $x = (-1, 0, \ldots, 0, 1)$ is one. Its values change only between the first and second positions and between the last two, so it has only two threshold cuts: vertex 1 against the rest, and vertex n against the rest. Each cuts only $n-1$ edges, fewer than $(n/2)^2$, and far fewer when n is large.

Trevisan's idea is to partition V into three sets L, C and R (left, center and right). The edges between L and R are cut, the vertices of C are left undecided, and the algorithm recurses on C.

By (21.1) and $x^\top D x = \sum_i d_i x_i^2$, the Courant–Fischer theorem for the problem $Px = \beta Dx$ (Lecture 9) gives

$$\beta_1 = \min_{x \neq 0} \frac{\sum_{(i,j)\in E} (x_i + x_j)^2}{\sum_i d_i x_i^2}.$$

We consider the ℓ_1 version of this quotient, restricted to vectors with entries in $\{0, \pm 1\}$:

$$\beta(G) = \min_{x \in \{0,\pm 1\}^n,\ x \neq 0} \frac{\sum_{(i,j)\in E} |x_i + x_j|}{\sum_i d_i |x_i|}. \tag{21.2}$$

Definition 21.6. For disjoint $X, Y \subseteq V$, let $E(X)$ be the set of edges with both endpoints in X, and let $E(X, Y)$ be the set of edges with one endpoint in X and the other in Y. For $S \subseteq V$, let

$$\partial S = \{(u, v) \in E \mid u \in S,\ v \notin S\} \qquad \text{and} \qquad \mathrm{vol}(S) = \sum_{i \in S} d_i.$$

For a partition $V = L \cup R \cup C$ into disjoint sets with $L \cup R \neq \emptyset$, let

$$\beta(L, R, C) = \frac{2|E(L)| + 2|E(R)| + |\partial(L \cup R)|}{\mathrm{vol}(L \cup R)}.$$

Given $x \in \{0, \pm 1\}^n$ with $x \neq 0$, let $L = \{i : x_i = 1\}$, $R = \{i : x_i = -1\}$ and $C = \{i : x_i = 0\}$. Then $|x_i + x_j| = 2$ on the edges of $E(L)$ and $E(R)$, $|x_i + x_j| = 1$ on the edges of $\partial(L \cup R)$, and $|x_i + x_j| = 0$ on the edges of $E(L, R)$ and $E(C)$, while $\sum_i d_i |x_i| = \mathrm{vol}(L \cup R)$. Hence the quotient in (21.2) equals $\beta(L, R, C)$, and

$$\beta(G) = \min\{\beta(L, R, C) \mid V = L \cup R \cup C \text{ a partition with } L \cup R \neq \emptyset\}.$$

Theorem 21.7 (Trevisan [Tre12])**.** *Let G be a graph with unit edge weights and no isolated vertices. Then*

$$\tfrac{1}{2}\beta_1 \leq \beta(G) \leq \sqrt{2\beta_1}.$$

The theorem does not require G to be connected. This matters because the algorithm below applies it to induced subgraphs.

Remark 21.8. Both bounds are effective. In the lower bound, every partition $V = L \cup R \cup C$ with $L \cup R \neq \emptyset$ gives a test vector for β_1, the vector x that is 1 on L, -1 on R and 0 on C; by the proof of (a) below, its Rayleigh quotient $\sum_{(i,j)\in E}(x_i + x_j)^2 / \sum_i d_i x_i^2$ is at most $2\beta(L, R, C)$. The upper bound needs only a vector with a small Rayleigh quotient: the proof below turns any $x \neq 0$ with $\sum_{(i,j)\in E}(x_i + x_j)^2 \leq \gamma \sum_i d_i x_i^2$ into a partition with $\beta(L, R, C) \leq \sqrt{2\gamma}$. As the random threshold t of that proof varies, the rounded vector changes only when t passes one of the values $x_i^2 / \max_j x_j^2$. Hence it suffices to try the positive values among them, of which there are at most n.

21.2 Max-Cut algorithm

Fix a constant $\varepsilon \in (0,1)$. For $C \subseteq V$, let $G[C]$ be the subgraph of G induced by C.

Algorithm 21.9 (Max-Cut algorithm)**. Input:** a graph $G = (V, E)$ with unit edge weights. **Output:** a partition of V into two sets.

MCA(G):

1) If $E = \emptyset$, return any partition of V. Otherwise, run the following steps on G with its isolated vertices removed, and add the isolated vertices to either side of the result.

2) Compute β_1 and an eigenvector x of $Px = \beta_1 Dx$.

3) If $\beta_1 \geq 2\varepsilon$, return a uniformly random partition. Otherwise:

   a) find a partition $V = L \cup R \cup C$ with $L \cup R \neq \emptyset$ and $\beta(L, R, C) \leq \sqrt{2\beta_1}$, as in the proof of Theorem 21.7;

   b) let $(L', R') = \text{MCA}(G[C])$, a partition of C;

   c) return the better of $(L \cup L', R \cup R')$ and $(L \cup R', R \cup L')$.

Since $L \cup R \neq \emptyset$, the recursive call is on fewer vertices, so the algorithm terminates.

21.3 Proof of Trevisan's theorem

We prove the two inequalities of Theorem 21.7: (a) $\frac{1}{2}\beta_1 \leq \beta(G)$ and (b) $\beta(G) \leq \sqrt{2\beta_1}$.

Proof of (a). Restricting the minimum that gives β_1 to nonzero vectors with entries in $\{0, \pm 1\}$ can only increase it. For such vectors $y_i^2 = |y_i|$, and $(y_i + y_j)^2 \leq 2|y_i + y_j|$, because $|y_i + y_j| \in \{0, 1, 2\}$ and $s^2 \leq 2s$ for $0 \leq s \leq 2$. Hence

$$\begin{aligned}
\beta_1 &= \min_{y \neq 0} \frac{\sum_{(i,j)\in E} (y_i + y_j)^2}{\sum_i d_i y_i^2} \\
&\leq \min_{y \in \{0,\pm 1\}^n,\ y \neq 0} \frac{\sum_{(i,j)\in E} (y_i + y_j)^2}{\sum_i d_i |y_i|} \\
&\leq \min_{y \in \{0,\pm 1\}^n,\ y \neq 0} \frac{2\sum_{(i,j)\in E} |y_i + y_j|}{\sum_i d_i |y_i|} = 2\beta(G).
\end{aligned}$$

□

For the upper bound we round a vector with entries in $[-1, 1]$ to one with entries in $\{0, \pm 1\}$. For $t \in (0, 1]$, define $r_t \colon [-1, 1] \to \{0, \pm 1\}$ by

$$r_t(z) = \begin{cases} 1 & \text{if } z \geq \sqrt{t}, \\ -1 & \text{if } z \leq -\sqrt{t}, \\ 0 & \text{otherwise.} \end{cases}$$

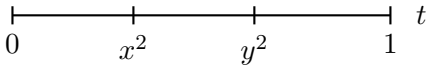
We pick t uniformly at random in $(0, 1]$, and write $r(z) = r_t(z)$ for the resulting random variable; all values $r(z)$ use the same t. Note that $r(-z) = -r(z)$, and that $|r(z)| = 1$ exactly when $t \leq z^2$.

Claim 21.10. *For all $x, y \in [-1, 1]$,*

$$\mathbb{E}\big[|r(x) + r(y)|\big] \leq |x + y| \cdot (|x| + |y|).$$

Proof. Both sides are unchanged if we swap x and y, and also if we replace (x, y) by $(-x, -y)$, because $r(-z) = -r(z)$. Hence we may assume that $x^2 \le y^2$ and $y \ge 0$.

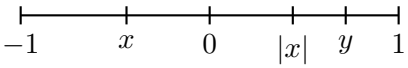
Case 1: $0 \le x \le y \le 1$. Then $r(x), r(y) \in \{0, 1\}$, with $r(x) = 1$ exactly when $t \le x^2$ and $r(y) = 1$ exactly when $t \le y^2$:



So $|r(x) + r(y)|$ is 2 if $t \le x^2$, it is 1 if $x^2 < t \le y^2$, and it is 0 otherwise. Since $xy \ge 0$,

$$\begin{aligned}\mathbb{E}\big[|r(x) + r(y)|\big] &= 2\Pr[t \le x^2] + 1 \cdot \Pr[x^2 < t \le y^2] \\ &= 2x^2 + (y^2 - x^2) = x^2 + y^2 \\ &\le (x + y)^2 = |x + y| \cdot (|x| + |y|).\end{aligned}$$

Case 2: $-1 \le x < 0 \le y \le 1$. The points lie as follows:



Now $r(x) \in \{0, -1\}$, with $r(x) = -1$ exactly when $t \le x^2$, and $r(y) \in \{0, 1\}$, with $r(y) = 1$ exactly when $t \le y^2$. Since $x^2 \le y^2$, the sum $r(x) + r(y)$ is nonzero exactly when $x^2 < t \le y^2$, and then it equals 1. Since $y + x \ge 0$ and $y - x = |y| + |x|$,

$$\begin{aligned}\mathbb{E}\big[|r(x) + r(y)|\big] &= \Pr[x^2 < t \le y^2] = y^2 - x^2 \\ &= (y + x)(y - x) = |x + y| \cdot (|x| + |y|).\end{aligned}$$

□

Proof of (b). Let x be an eigenvector for β_1, so that by (21.1)

$$\sum_{(i,j)\in E} (x_i + x_j)^2 = \beta_1 \sum_i d_i x_i^2.$$

Scale x so that $\max_i |x_i| = 1$; then every $r(x_i)$ is defined. By the definition of r, $\mathbb{E}|r(x_i)| = \Pr[t \le x_i^2] = x_i^2$. We use this in the last step below. By Claim 21.10 and linearity of expectation, then the Cauchy–Schwarz inequality, then $(a + b)^2 \le 2(a^2 + b^2)$, and finally $\sum_{(i,j)\in E}(x_i^2 + x_j^2) = \sum_i d_i x_i^2$,

$$\begin{aligned}\mathbb{E}\Big[\sum_{(i,j)\in E} |r(x_i) + r(x_j)|\Big] &\le \sum_{(i,j)\in E} |x_i + x_j| \cdot (|x_i| + |x_j|) \\ &\le \Big(\sum_{(i,j)\in E} (x_i + x_j)^2\Big)^{1/2} \Big(\sum_{(i,j)\in E} (|x_i| + |x_j|)^2\Big)^{1/2} \\ &\le \Big(\beta_1 \sum_i d_i x_i^2\Big)^{1/2} \Big(2 \sum_{(i,j)\in E} (x_i^2 + x_j^2)\Big)^{1/2} \\ &= \sqrt{2\beta_1} \sum_i d_i x_i^2 = \sqrt{2\beta_1}\, \mathbb{E}\Big[\sum_i d_i |r(x_i)|\Big].\end{aligned}$$

For $t \in (0, 1]$, let $N_t = \sum_{(i,j)\in E} |r_t(x_i) + r_t(x_j)|$ and $M_t = \sum_i d_i |r_t(x_i)|$. Choose i^* with $|x_{i^*}| = 1$. Then $r_t(x_{i^*}) \ne 0$ for every t, so $M_t \ge d_{i^*} > 0$. The display says that $\mathbb{E}[N_t - \sqrt{2\beta_1}\, M_t] \le 0$, so some t has $N_t \le \sqrt{2\beta_1}\, M_t$. For this t, the vector $(r_t(x_1), \dots, r_t(x_n))$ is nonzero, has entries in $\{0, \pm 1\}$, and its quotient in (21.2) is $N_t / M_t \le \sqrt{2\beta_1}$. Hence $\beta(G) \le \sqrt{2\beta_1}$. □

21.4 Analysis of the algorithm

In this section we measure cuts by the objective of Definition 21.2, which counts every cut edge twice. For a graph G with unit edge weights, let $\mathrm{Opt}(G) = 2\,\mathrm{MaxCut}(G)$ be the maximum of the objective, and let $\mathrm{Alg}(G)$ be the objective value of the partition returned by $\mathrm{MCA}(G)$, a random variable. A uniformly random partition has expected objective value $|E|$, since each edge is cut with probability $\frac{1}{2}$ and then counts 2.

Lemma 21.11. *Let G have unit edge weights, at least one edge, and no isolated vertices. Then*

$$\mathrm{Opt}(G) \le (1 - \tfrac{1}{2}\beta_1)\, 2|E|.$$

In particular, if $\mathrm{Opt}(G) \ge (1-\varepsilon)\,2|E|$, then $\beta_1 \le 2\varepsilon$; and if $\beta_1 \ge 2\varepsilon$, then $\mathrm{Opt}(G) \le (1-\varepsilon)\,2|E|$.

Proof. Let (L, R) be an optimal cut, so that $\mathrm{Opt}(G) = 2|E(L,R)|$, and view it as a partition with $C = \emptyset$. Then $\partial(L \cup R) = \emptyset$ and $\mathrm{vol}(L \cup R) = \mathrm{vol}(V) = 2|E|$. The edges not cut are those of $E(L)$ and $E(R)$, and their number is $|E| - |E(L,R)|$. Hence

$$\beta(G) \le \beta(L, R, \emptyset) = \frac{2|E(L)| + 2|E(R)|}{2|E|} = 1 - \frac{\mathrm{Opt}(G)}{2|E|}.$$

By Theorem 21.7, $\beta_1 \le 2\beta(G)$, and the inequality follows. Both consequences are immediate. □

Theorem 21.12. *Let $0 < \varepsilon < 1$ and $\rho = \min\{1 - 2\sqrt{\varepsilon},\ \frac{1}{2(1-\varepsilon)}\}$. For every graph G with unit edge weights,*

$$\mathbb{E}[\mathrm{Alg}(G)] \ge \rho \cdot \mathrm{Opt}(G).$$

For $\varepsilon = 0.0554$ we have $\rho > 0.529$.

Proof. We use induction on the number of vertices. If G has no edges, then $\mathrm{Opt}(G) = 0$ and there is nothing to prove. Isolated vertices change neither $\mathrm{Alg}(G)$ nor $\mathrm{Opt}(G)$, so we may assume that G has none.

Case 1: $\beta_1 \ge 2\varepsilon$. The algorithm returns a uniformly random partition, so $\mathbb{E}[\mathrm{Alg}(G)] = |E|$. By Lemma 21.11, $\mathrm{Opt}(G) \le (1-\varepsilon)\,2|E|$, hence

$$\mathbb{E}[\mathrm{Alg}(G)] = |E| \ge \frac{\mathrm{Opt}(G)}{2(1-\varepsilon)} \ge \rho \cdot \mathrm{Opt}(G).$$

Thus in this case the algorithm is a $\frac{1}{2(1-\varepsilon)}$-approximation.

Case 2: $\beta_1 < 2\varepsilon$. By the upper bound of Theorem 21.7, the algorithm finds a partition L, R, C with

$$\beta(L, R, C) \le \sqrt{2\beta_1} \le 2\sqrt{\varepsilon}.$$

Let $S = L \cup R$, and let $(L', R') = \mathrm{MCA}(G[C])$, so that $L' \cup R' = C$. The algorithm compares the two candidates $(L \cup L', R \cup R')$ and $(L \cup R', R \cup L')$. Every edge of $E(L,R)$ is cut by both. An edge of $E(C)$ is cut by both if (L', R') cuts it, and by neither otherwise. The edges of $E(L)$ and $E(R)$ are cut by neither. Every edge of $E(S,C)$ is cut by exactly one of the two: for example, an edge between L and L' is cut only by the second candidate, and an edge between L and R' only by the first. Hence the objective values of the two candidates add up to $2\big(2|E(L,R)| + |E(S,C)|\big) + 2\,\mathrm{Alg}(G[C])$, and the better one has at least half of this:

$$\mathrm{Alg}(G) \ge X + \mathrm{Alg}(G[C]), \qquad X = 2|E(L,R)| + |E(S,C)|.$$

On the other hand, an optimal cut of G restricted to C is a cut of $G[C]$, and the edges outside $E(C)$ contribute at most 2 each, so

$$\mathrm{Opt}(G) \le Y + \mathrm{Opt}(G[C]), \qquad Y = 2|E(L)| + 2|E(R)| + 2|E(L,R)| + 2|E(S,C)|.$$

We now show that $X \ge (1-2\sqrt{\varepsilon})\,Y$. Since $\mathrm{vol}(S) = 2|E(L)|+2|E(R)|+2|E(L,R)|+|E(S,C)|$ and $\partial S = E(S,C)$,

$$2\sqrt{\varepsilon} \ge \beta(L,R,C) = \frac{2|E(L)| + 2|E(R)| + |E(S,C)|}{\mathrm{vol}(S)} = 1 - \frac{2|E(L,R)|}{\mathrm{vol}(S)}.$$

Hence

$$1 - 2\sqrt{\varepsilon} \le q := \frac{2|E(L,R)|}{\mathrm{vol}(S)} = \frac{|E(L,R)|}{|E(L)| + |E(R)| + |E(L,R)| + \frac{1}{2}|E(S,C)|}.$$

Here $\mathrm{vol}(S) > 0$ because $S \neq \emptyset$, and $q \le 1$. If $a/b \le 1$ with $b > 0$ and $c \ge 0$, then $(a+c)/(b+c) \ge a/b$; adding $\frac{1}{2}|E(S,C)|$ to the numerator and the denominator of q therefore gives

$$\frac{X}{Y} = \frac{|E(L,R)| + \frac{1}{2}|E(S,C)|}{|E(L)| + |E(R)| + |E(L,R)| + |E(S,C)|} \ge q \ge 1 - 2\sqrt{\varepsilon} \ge \rho.$$

Finally, C has fewer vertices than G, so the induction hypothesis gives $\mathbb{E}[\mathrm{Alg}(G[C])] \ge \rho \cdot \mathrm{Opt}(G[C])$. Since X and Y are determined by G,

$$\mathbb{E}\big[\mathrm{Alg}(G)\big] \ge X + \mathbb{E}\big[\mathrm{Alg}(G[C])\big] \ge \rho Y + \rho \cdot \mathrm{Opt}(G[C]) \ge \rho \cdot \mathrm{Opt}(G).$$

For $\varepsilon = 0.0554$, we have $1 - 2\sqrt{\varepsilon} = 0.52926\ldots$ and $\frac{1}{2(1-\varepsilon)} = 0.52932\ldots$, so $\rho > 0.529$. □

The two terms of ρ are nearly equal at $\varepsilon = 0.0554$: a larger ε improves the random-cut case and worsens the recursive case. A better ratio, 0.614, is known. Soto [Sot15] showed that Trevisan's algorithm achieves it.

22 Solving Graph Laplacians in Nearly $O(m \log n)$ Time

Lecture of March 30, 2020. Based on Gary L. Miller's handwritten notes.

This lecture outlines a solver for Laplacian systems $L_G x = b$, due to Koutis, Miller and Peng [KMP11], whose running time is $\widetilde{O}(m \log n \log(1/\varepsilon))$. It combines preconditioned iterative methods and low-stretch spanning trees (Lectures 17 and 18) with sampling by effective resistances and the Ahlswede–Winter theorem (Lecture 8). The solver builds a chain of graphs, each a good preconditioner for the one before it. Phase 1 scales up a low-stretch spanning tree and samples the remaining edges, and Phase 2 repeats this with a constant scaling factor. The key observation is that when samples are not combined, every sampled edge has the same small stretch, and this makes the number of nontree edges drop by a constant factor at each level.

22.1 Nearly $O(m \log n)$ Laplacian solver

Throughout this lecture, $G = (V, E, w)$ is a connected graph with n vertices, m edges and positive edge weights; parallel edges are allowed. We identify a graph with its Laplacian. Thus $G \preceq H$ means $L_G \preceq L_H$. The sum $G + H$ of two graphs on the same vertex set is the graph with Laplacian $L_G + L_H$, so that weights of common edges add, and for $c > 0$ the graph cG has all weights multiplied by c. We write $H \subseteq H'$ if every edge weight of H is at most the corresponding weight in H'; then $H \preceq H'$. For a vector x and a symmetric positive semidefinite (SPSD) matrix A, $\|x\|_A = \sqrt{x^\top A x}$.

Theorem 22.1. *There is a randomized algorithm that, given the Laplacian $A = L_G$ of G, a vector b in the range of A, and $\varepsilon \in (0, 1)$, returns a vector x with*

$$\|x - \bar{x}\|_A \le \varepsilon \|\bar{x}\|_A$$

for every solution $\bar{x}$ of $A\bar{x} = b$, in expected time $\widetilde{O}\big(m \log n \log(1/\varepsilon)\big)$.

Both sides of the inequality are the same for every solution $\bar{x}$, since two solutions differ by a multiple of $\mathbf{1}$ and $A\mathbf{1} = 0$. The notation $\widetilde{O}$ hides factors that are powers of $\log \log n$. We do not prove the theorem in full: the rest of the lecture describes the algorithm and proves its main estimates.

For a spanning tree T of G and an edge $e = (u, v)$ of weight w_e, let $\mathrm{ER}_T(e) = \mathrm{ER}_T(u, v)$ be the effective resistance between u and v in T. It is the sum of $1/w_f$ over the edges f of the tree path from u to v. The *stretch* of e is $\mathrm{stretch}_T(e) = w_e \, \mathrm{ER}_T(e)$, as in Lecture 18. For a set F of edges, $\mathrm{stretch}_T(F) = \sum_{e \in F} \mathrm{stretch}_T(e)$, and $\mathrm{stretch}_T(G) = \mathrm{stretch}_T(E(G))$. We need the following result, which we do not prove. It is from [KMP11], which speeds up the construction of Abraham, Bartal and Neiman [ABN08].

Theorem 22.2. *Every graph G as above has a spanning tree T, a* low-stretch spanning tree *(LSST), with*

$$\mathrm{stretch}_T(G) = O\big(m \log n \, (\log \log n)^3\big),$$

and such a tree can be computed in time $O\big(m \log n + n \log n \log \log n\big)$.

We also use the Ahlswede–Winter theorem (AW) of Lecture 8; Lecture 11 derives a version of it from Tropp's matrix Chernoff bound.

Theorem 22.3 (Ahlswede–Winter)**.** *Let $Y_1, \dots, Y_k$ be independent, identically distributed random SPSD $n \times n$ matrices with $\mathbb{E}[Y_i] = Z$, and let $\mu > 0$ be such that $Y_i \preceq \mu Z$ for every possible value of Y_i. Then for every $0 < \varepsilon \le 1$,*

$$\Pr\Big[(1-\varepsilon)Z \preceq \frac{1}{k}\sum_{i=1}^{k} Y_i \preceq (1+\varepsilon)Z\Big] \ge 1 - 2n\, e^{-\varepsilon^2 k/(4\mu)}.$$

22.2 High-level view of the algorithm

The algorithm has two parts.

1) Given $G = (V, E)$, it generates a *solver chain* of graphs

$$G = G_0, G_1, \dots, G_\ell,$$

with $\ell = O(\log m)$ (which is $O(\log n)$ when G has no parallel edges) and G_ℓ the graph at which Phase 2 below stops, which has fewer than $\sqrt{n}$ nontree edges, such that the relative condition numbers satisfy

$$\kappa(G_0, G_1) = O(\log^2 n) \qquad \text{and} \qquad \kappa(G_i, G_{i+1}) = O(1) \quad \text{for } 1 \le i \le \ell - 1.$$

Here $\kappa(G, H) = \sigma(G/H)\,\sigma(H/G)$, where $\sigma(G/H)$ is the least τ with $G \preceq \tau H$ (Definition 17.6).

2) It runs *recursive preconditioned Chebyshev* (RPC) on the sequence $G_1, \dots, G_\ell$: each iteration for a system $L_{G_i} y_i = b_i$ makes calls that solve systems $L_{G_{i+1}} y_{i+1} = b_{i+1}$.

22.3 The solver has two distinct phases

In Phase 1 we reduce solving $L_G x = b$ to solving systems in a spine-heavy graph.

Definition 22.4. Fix a constant $c_0 > 0$. A graph G with n vertices and m edges is *spine-heavy* if it has a spanning tree T with

$$\operatorname{stretch}_T\big(E(G) \setminus E(T)\big) \le c_0\, m / \log n.$$

Algorithm 22.5 (Phase1(G, b))**.**

1) Compute an LSST T of G (Theorem 22.2).

2) Set $H = G + (t-1)T$, where $t = c \log^2 n$ for a suitable constant c. The tree of H is tT.

3) Set $H' = \text{Sample}(H, tT)$, which samples the nontree edges (Algorithm 22.6).

4) Run the preconditioned conjugate gradient method $\text{PCG}(G, H')$ on $L_G x = b$ with preconditioner $L_{H'}$, solving each system $L_{H'} y = b'$ by a call $\text{Phase2}(H', tT, b')$.

5) Perform $O(\sqrt{t}) = O(\log n)$ iterations of step 4 per bit of accuracy required.

In the solver chain, $G_1 = H'$. All later calls use recursive preconditioned Chebyshev.

AW samples with replacement. In Lecture 8 we combined identical samples into one edge. Here we do not combine samples, so the sampled graph is a multigraph. In the theorem of Spielman and Srivastava [SS11], edge e is sampled with probability $w_e \operatorname{ER}_e / (n-1)$, where ER_e is its effective resistance in G. More generally, we recall the following from Lecture 8: given numbers $r_e \ge w_e \operatorname{ER}_e$, we can sample with probabilities r_e / R, where $R = \sum_e r_e$, a total of $C\, R \log n$ times, for a suitable constant C; with high probability, this gives a spectral sparsifier with $O(R \log n)$ edges.

22.4 Sampling nontree edges

Algorithm 22.6 (Sample(G,T))**. Input:** a weighted graph $G=(V,E,w)$ and a spanning tree T of G, with the weights it has in G. The constant C_s and the parameter $\delta \in (0,1)$ are fixed.

1) For each $e \in E(G)\setminus E(T)$, set $p'_e = w_e \operatorname{ER}_T(e)$.
2) Set $s = \sum_{e\in E(G)\setminus E(T)} p'_e = \operatorname{stretch}_T\big(E(G)\setminus E(T)\big)$.
3) Set $p_e = p'_e/s$.
4) Set $N = \lceil C_s\, s \log n \log(1/\delta)\rceil$.
5) Sample N edges $e_1,\dots,e_N$ of $E(G)\setminus E(T)$ independently, each equal to e with probability p_e, and give a sampled copy of e the weight w_e/p_e.
6) Return the multigraph
$$T + \frac{w_{e_1}}{N p_{e_1}}\, e_1 + \cdots + \frac{w_{e_N}}{N p_{e_N}}\, e_N,$$
where $\gamma\, e$ denotes a copy of the edge e with weight γ.

To apply AW, we view a sample as the random graph

$$Y_i = T + \frac{w_e}{p_e}\, e \quad \text{with probability } p_e \qquad \big(e \in E(G)\setminus E(T)\big).$$

Then $\mathbb{E}[Y_i] = T + \sum_e p_e \frac{w_e}{p_e}\, e = G$, and $\frac{1}{N}\sum_{i=1}^N Y_i$ is the graph returned in step 6.

Lemma 22.7. *Let $H = \text{Sample}(G,T)$, with s and N as in Algorithm 22.6. Every possible value of Y_i satisfies $Y_i \preceq (1+s)\,G$, and*

$$\Pr\big[\tfrac{1}{2}G \preceq H \preceq \tfrac{3}{2}G\big] \ge 1 - 2n\, e^{-N/(16(1+s))}.$$

If $s \ge 1$, $C_s \ge 64$ and $\delta \le 1/e$, this probability is at least $1 - 2/n$.

Proof. Fix a nontree edge $e=(u,v)$, and let L_e be the Laplacian of the single edge (u,v) with weight 1. As in Lecture 8, $L_e \preceq \operatorname{ER}_G(e)\, L_G$, where $\operatorname{ER}_G(e)$ is the effective resistance between u and v in G. Since T is a subgraph of G, Rayleigh's monotonicity law (Lecture 3) gives $\operatorname{ER}_G(e) \le \operatorname{ER}_T(e)$. Hence

$$\frac{w_e}{p_e} L_e \preceq \frac{w_e \operatorname{ER}_T(e)}{p_e}\, G = \frac{p'_e}{p_e}\, G = s\, G.$$

Since also $T \preceq G$, every value of Y_i satisfies $Y_i \preceq (1+s)G$. The matrices Y_i are SPSD and independent, with $\mathbb{E}[Y_i] = G$, so AW with $Z = G$, $\mu = 1+s$, $\varepsilon = \frac{1}{2}$ and $k = N$ gives the probability bound. If $s \ge 1$, then $s/(1+s) \ge \frac{1}{2}$, so with natural logarithms

$$\frac{N}{16(1+s)} \ge \frac{C_s\, s \log n \log(1/\delta)}{16(1+s)} \ge \frac{C_s}{32} \log n \log(1/\delta) \ge 2 \log n$$

when $C_s \ge 64$ and $\log(1/\delta) \ge 1$. Then $2n\, e^{-N/(16(1+s))} \le 2n \cdot n^{-2} = 2/n$. □

Thus the factor $\log(1/\delta)$ in N controls the failure probability: for $s \geq 1$, the proof gives the bound $2n \cdot n^{-(C_s/32)\log(1/\delta)}$, which decreases as δ decreases. In the solver, δ is a constant, so that $N = O(1 + s \log n)$.

In Phase 1, the graph $H = G + (t-1)T$ has the tree tT. Multiplying the tree weights by t divides every tree resistance by t, so by Theorem 22.2

$$\operatorname{stretch}_{tT}(E(H) \setminus E(T)) = \frac{1}{t} \operatorname{stretch}_T(E(G) \setminus E(T)) = O\Big(\frac{m\,(\log\log n)^3}{\log n}\Big) = \widetilde{O}(m/\log n),$$

and the number of samples is $N = \widetilde{O}\big(m \log(1/\delta)\big)$.

22.5 Graph algorithms on trees

Many problems on trees can be solved in time $O(n)$. The one we need is this: given a tree T and nontree edges $e_1, \ldots, e_m$, compute $\mathrm{ER}_T(e_1), \ldots, \mathrm{ER}_T(e_m)$. These give the probabilities in Algorithm 22.6.

Root T at a vertex r. For every vertex a, let $\rho(a) = \mathrm{ER}_T(r, a)$. A breadth-first search from r computes all these labels in time $O(n)$, since $\rho(a) = \rho(a') + 1/w(a', a)$ when a' is the parent of a.

Definition 22.8. For vertices a and b of the rooted tree T, the *lowest common ancestor* $\mathrm{LCA}(a, b)$ is the vertex farthest from r that lies on both tree paths from r to a and from r to b.

Lemma 22.9. *For all $a, b \in V(T)$,*

$$\mathrm{ER}_T(a, b) = \rho(a) + \rho(b) - 2\,\rho(\mathrm{LCA}(a, b)).$$

Proof. Let $z = \mathrm{LCA}(a, b)$. The tree path from a to b consists of the path from a up to z and the path from z down to b. Resistances in series add, and the path from r to a passes through z, so the first part has resistance $\rho(a) - \rho(z)$. Similarly, the second part has resistance $\rho(b) - \rho(z)$. □

Theorem 22.10 (Tarjan [Tar79])**.** *Given a rooted tree with n vertices and m pairs of vertices, the lowest common ancestors of all pairs can be computed in time $O\big((n+m)\,\alpha(n+m)\big)$, where α is the inverse Ackermann function.*

We use this theorem without proof. Together with Lemma 22.9, it computes all the tree resistances $\mathrm{ER}_T(e_1), \ldots, \mathrm{ER}_T(e_m)$ within the same time.

22.6 Timing analysis of Phase 1 assuming Phase 2

Claim 22.11. *Suppose that Phase 2 solves the systems $L_{H'}y = b'$ exactly, and that $\frac{1}{2}H \preceq H' \preceq \frac{3}{2}H$, which by Lemma 22.7 holds with probability at least $\frac{1}{2}$ when $s \geq 1$, $C_s \geq 64$, $\delta \leq 1/e$ and $n \geq 4$. Then $O(\log n)$ iterations of step 4 of Phase 1 reduce the error of $L_G x = b$ by a constant factor.*

Proof. Since $T \preceq G$, we have $G \preceq H = G + (t-1)T \preceq tG$. Hence

$$\sigma(G/H) \leq 1, \qquad \sigma(H/G) \leq t = c\log^2 n, \qquad \text{so} \qquad \kappa(G, H) \leq c\log^2 n.$$

Next, $\frac{1}{2}H \preceq H' \preceq \frac{3}{2}H$ gives $\sigma(H/H') \leq 2$ and $\sigma(H'/H) \leq \frac{3}{2}$, so $\kappa(H, H') \leq 3$. Since $\sigma(G/H') \leq \sigma(G/H)\,\sigma(H/H')$ and $\sigma(H'/G) \leq \sigma(H'/H)\,\sigma(H/G)$,

$$\kappa(G, H') \leq \kappa(G, H)\,\kappa(H, H') \leq 3 \cdot c\log^2 n = O(\log^2 n).$$

PCG needs only $O\big(\sqrt{\kappa(G, H')}\big)$ iterations per bit of accuracy (Lectures 16 and 17), so this step needs only $O(\log n)$ iterations. □

22.7 The critical trick to get a one-log solver

Claim 22.12. *Let $\hat{e}$ be a sampled copy of a nontree edge e in the graph returned by* Sample(G,T)*, with its normalized weight $w_{\hat{e}} = \frac{w_e}{Np_e}$. Then*

$$\text{stretch}_T(\hat{e}) = \frac{s}{N} \le \frac{1}{C_s \log n \log(1/\delta)}.$$

In particular, all sampled edges have the same stretch, whatever their original weights.

Proof. We have $p_e = p'_e/s = w_e \,\text{ER}_T(e)/s$, so

$$w_{\hat{e}} = \frac{w_e}{p_e} \cdot \frac{1}{N} = \frac{w_e\, s}{w_e \,\text{ER}_T(e)} \cdot \frac{1}{N} = \frac{s}{N \,\text{ER}_T(e)},$$

where the factor w_e cancels. Hence the stretch of $\hat{e}$ is $w_{\hat{e}} \,\text{ER}_T(e) = s/N$. Finally, the choice of N in Algorithm 22.6 gives $N \ge C_s\, s \log n \log(1/\delta)$, that is, $s/N \le 1/(C_s \log n \log(1/\delta))$. □

22.8 Phase 2

Fix a constant $k > 1$.

Algorithm 22.13 (Phase2(H,T,b))**. Input:** a multigraph H, a spanning tree T of H with the weights it has in H, and a vector b.

As long as H has at least $\sqrt{n}$ nontree edges, Phase2 does the following:

1) Set $H' = H + (k-1)T$, which multiplies the weights of the tree edges by k; note that $kT \subseteq H'$.

2) Set $H'' = \text{Sample}(H', kT)$.

3) Solve $L_H x = b$ by preconditioned Chebyshev iteration Pre-Cheb(H, H'') with preconditioner $L_{H''}$,

4) solving each system $L_{H''} y = b''$ by a call Phase2(H'', kT, b'').

We do not specify the base case, that is, how Phase2 solves $L_H x = b$ once H has fewer than $\sqrt{n}$ nontree edges.

22.9 Analysis of Phase 2

Claim 22.14. *In Phase 2, if $\frac{1}{2}H' \preceq H'' \preceq \frac{3}{2}H'$, then $\kappa(H, H'') \le 3k$. By Lemma 22.7, applied to* Sample(H', kT)*, the hypothesis holds with high probability, namely with probability at least $1 - 2/n$ if the nontree edges of H' have total stretch at least 1 with respect to kT, $C_s \ge 64$ and $\delta \le 1/e$.*

Proof. First, $H \preceq H' \preceq kH$, since $H \subseteq H' \subseteq kH$. Combined with the hypothesis,

$$\tfrac{1}{2}H \preceq \tfrac{1}{2}H' \preceq H'' \preceq \tfrac{3}{2}H' \preceq \tfrac{3}{2}kH,$$

so $\sigma(H/H'') \le 2$ and $\sigma(H''/H) \le \frac{3}{2}k$, that is, $\kappa(H, H'') \le 3k$. □

Claim 22.15. *Suppose that the nontree edges of H are m sampled copies, each of stretch at most $1/(C_s \log n \log(1/\delta))$ with respect to T, as Claim 22.12 guarantees for the output of* Sample. *Then $H'' = \text{Sample}(H', kT)$ has at most $\lceil m/k \rceil$ nontree edges.*

Proof. The number of samples is $N' = \lceil C_s \operatorname{stretch}_{kT}(E(H') \setminus E(T)) \log n \log(1/\delta) \rceil$. Since we dilated T by k, every tree resistance is divided by k, so

$$\operatorname{stretch}_{kT}(E(H') \setminus E(T)) = \frac{1}{k} \operatorname{stretch}_T(E(H) \setminus E(T)) \le \frac{m}{k\, C_s \log n \log(1/\delta)}.$$

Hence

$$N' \le \left\lceil C_s \cdot \frac{m}{k\, C_s \log n \log(1/\delta)} \cdot \log n \log(1/\delta) \right\rceil = \left\lceil \frac{m}{k} \right\rceil. \qquad \square$$

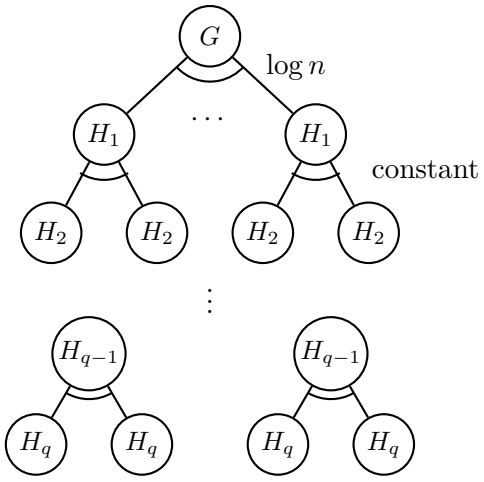
22.10 In summary

Phase 1 reduces solving $L_G x = b$ to $O(\log n)$ calls per bit of accuracy that solve systems $L_{H'} x' = b'$, all with the same graph H', and H' is spine-heavy. Indeed, by Claim 22.12 each of the N nontree edges of H' has stretch at most $1/(C_s \log n \log(1/\delta))$ with respect to its tree tT, so their total stretch is at most $|E(H')|/(C_s \log(1/\delta) \log n)$. This is the bound of Definition 22.4 with $c_0 = 1/(C_s \log(1/\delta))$.

Phase 2, with $k = 3$, generates a chain of graphs $H_1 = H', H_2, \ldots, H_q$, all containing the same tree edges $E(T)$, such that

1) $|E(H_{i+1}) \setminus E(T)| \le \lceil |E(H_i) \setminus E(T)|/3 \rceil$, by Claim 22.15;
2) $\kappa(H_i, H_{i+1}) \le 9$, with high probability, by Claim 22.14;
3) to solve $L_{H_i} x = b$, we make a constant number of calls that solve systems $L_{H_{i+1}} x' = b'$, since preconditioned Chebyshev iteration needs $O(\sqrt{\kappa(H_i, H_{i+1})}) = O(1)$ iterations per constant factor of error reduction (Lectures 14 and 17).

The calls form the following tree: per bit of accuracy, the top level makes $O(\log n)$ calls on H_1, and every later level makes a constant number of calls on the next graph.



The vertex count, however, is *not* decreasing: every H_i contains the spanning tree T on all n vertices. Since the number of calls grows by a constant factor at each level, the total work stays small only if the cost per call decreases faster than the number of calls grows. The next subsection deals with the vertices.

22.11 Removing degree one and two nodes using Gaussian elimination

Theorem 22.16. *Let G be a multigraph with $n \geq 2$ vertices, let T be a spanning tree of G, and let m be the number of nontree edges. If no vertex of G has degree one or two, counting parallel edges with multiplicity, then $n \leq 2m$.*

Proof. Let n_1, n_2 and n_3 be the numbers of vertices of degree 1 (the leaves), degree 2, and degree at least 3 in T. Then $n_1 \geq n_3 + 2$. Indeed, since T is a tree with $n \geq 2$ vertices, it has no vertex of degree 0, and its degrees sum to $2(n-1)$, so $\sum_v (\deg_T(v) - 2) = -2$. Hence $n_1 = 2 + \sum_{\deg_T(v) \geq 3} (\deg_T(v) - 2) \geq n_3 + 2$. Thus

$$n = n_1 + n_2 + n_3 \leq 2n_1 + n_2.$$

We next count the endpoints of nontree edges. Since T spans G and $n \geq 2$, every vertex has degree at least 1 in G, hence at least 3. So each leaf of T is an endpoint of at least 2 nontree edges, and each vertex of degree 2 in T is an endpoint of at least one. Every nontree edge has two endpoints, so $2m \geq 2n_1 + n_2 \geq n$. □

The full algorithm pivots out vertices of degree one and two by Gaussian elimination after each level of Phase 2. When no such vertices remain, Theorem 22.16 bounds the number of vertices by twice the number of nontree edges, so the vertex count decreases along with the nontree edges (Lecture 23).

23 Components of the Fast Laplacian Solver (Solver Chain)

Lecture of April 1, 2020. Based on Gary L. Miller's handwritten notes.

This lecture takes apart the solver of Lecture 22 and explains why each component is there. A recursive solver needs a chain of graphs in which every graph is a good preconditioner for the previous one and is much cheaper to solve. Sparsification alone cannot produce such a chain, because a sparsifier of a connected graph keeps at least $n-1$ edges. Instead we keep a spanning tree, scale it up, and sample only the nontree edges, using the tree to estimate their resistances. When samples are not combined, every sampled edge has the same small stretch, and the number of nontree edges then falls geometrically along the chain. Finally, we eliminate vertices of degree one and two, so that the number of vertices falls as well, and we check that elimination keeps the tree low stretch.

23.1 Components of a fast Laplacian solver

Throughout this lecture, graphs are connected and have positive edge weights, and parallel edges are allowed. We use the conventions of Lecture 22: a graph is identified with its Laplacian, so $G \preceq H$ means $L_G \preceq L_H$, the graph $G+H$ has Laplacian $L_G + L_H$, and cT is T with all weights multiplied by $c > 0$. The relative condition number is $\kappa(G,H) = \sigma(G/H)\,\sigma(H/G)$, where $\sigma(G/H)$ is the least τ with $G \preceq \tau H$ (Lecture 17). The stretch $\mathrm{stretch}_T$ and the tree resistance ER_T are as in Lecture 22.

Our goal is a *recursive* iterative solver: given a graph G and a vector b, solve $L_G x = b$ approximately. The idea is to construct a sequence of graphs $G = G_0, G_1, \ldots, G_\ell$, a *solver chain*, such that

1) $\kappa(G_i, G_{i+1})$ is small;

2) solving systems in $L_{G_{i+1}}$ is much easier than solving systems in L_{G_i}, in the sense that

$$\#\mathrm{iterations}(G_i, G_{i+1}) \le \mathrm{Size}(G_i)/\mathrm{Size}(G_{i+1}),$$

where $\#\mathrm{iterations}(G_i, G_{i+1})$ is the number of iterations that a solve in G_i performs with preconditioner G_{i+1}, and $\mathrm{Size}(G_i)$ is the number of vertices plus the number of edges of G_i.

Remark 23.1. Condition 2 keeps the total work under control. Write q_i for $\#\mathrm{iterations}(G_i, G_{i+1})$. A solve in G_i costs $O(\mathrm{Size}(G_i))$ per iteration plus one solve in G_{i+1} per iteration, so level j of the chain is entered $q_0 \cdots q_{j-1}$ times. By condition 2, $q_0 \cdots q_{j-1} \le \mathrm{Size}(G_0)/\mathrm{Size}(G_j)$, so the work at level j, not counting the recursive calls, is $O(q_j \mathrm{Size}(G_0))$. The total work is therefore $O\big(\mathrm{Size}(G_0) \sum_j q_j\big)$.

A first goal is to make the edge count decrease along the chain. By the theorem of Spielman and Srivastava [SS11] (Lecture 8), every graph has a spectral sparsifier with $m = O(n \log n)$ edges. If we did not care about the time to construct the solver chain, the sparsifiers of Batson, Spielman and Srivastava [BSS12] would get us down to $m = O(n)$. Here sparsification gets stuck. We would like a sparsifier H with $m < n-1$ edges, but then H is not even connected. Then the kernel of L_H has dimension at least 2, while the kernel of L_G is spanned by $\mathbf{1}$, so no τ satisfies $G \preceq \tau H$.

23.2 Ultra sparsifiers

Definition 23.2. Let G be a graph with n vertices, and let $\kappa \geq 1$. A pair (H, T) is an *ultra-sparsifier* of G with condition number at most κ if

1) $V(H) = V(G)$;

2) T is a spanning tree of H;

3) $\kappa(G, H) \leq \kappa$;

4) $|E(H) \setminus E(T)| = o(n)$.

Condition 4, and the requirement that κ be small, refer to a family of graphs with $n \to \infty$.

Lemma 23.3. *Let T be a spanning tree of G and let $k \geq 1$. Then*

1) $k \cdot \text{stretch}_{kT}(E(G) \setminus E(T)) = \text{stretch}_T(E(G) \setminus E(T))$;

2) $H = G + (k-1)T$ *satisfies* $G \preceq H \preceq kG$, *so* $\kappa(G, H) \leq k$.

Proof. Multiplying every tree weight by k divides every tree resistance by k, so $\text{ER}_{kT}(e) = \text{ER}_T(e)/k$ for every edge e; this gives part 1. For part 2, $H - G = (k-1)T \succeq 0$, and $kG - H = (k-1)(G - T) \succeq 0$ because $G - T$ is the graph of the nontree edges. Hence $\sigma(G/H) \leq 1$ and $\sigma(H/G) \leq k$. □

Until now we have sampled all edges. For these solvers we sample only the nontree edges, and we use the tree to estimate their resistances. The tree resistances are upper estimates: $\text{ER}_G(e) \leq \text{ER}_T(e)$ by Rayleigh's monotonicity law (Lecture 3), since T is a subgraph of G. This is what makes the sampling of Lecture 22 work.

Let $n \geq 3$, so that $\log^2 n \geq 1$, and suppose that T is a low-stretch spanning tree with $\text{stretch}_T(E(G) \setminus E(T)) = m \log n$; this is the bound of Lecture 22 without its $(\log\log n)^3$ factor. Consider

$$H' = G + (\log^2 n - 1)\, T, \qquad T' = (\log^2 n)\, T,$$

so that T' is the tree of H'. By Lemma 23.3 with $k = \log^2 n$,

1) $\text{stretch}_{T'}(E(H') \setminus E(T)) = m \log n / \log^2 n = m / \log n$;

2) $\kappa(G, H') \leq \log^2 n$.

If we now sample the nontree edges of H', using resistances estimated with T', that is, if we let $H = \text{Sample}(H', T')$ as in Lecture 22, we get

1) $\lceil C_s (m/\log n) \log n \log(1/\delta) \rceil = O(m)$ nontree edges, since δ is a constant;

2) $\kappa(H', H) \leq 3$ with probability at least $\frac{1}{2}$ (Lecture 22), under the conditions on s, C_s, δ and n in Claim 22.11, with $s = m/\log n$.

Combining tree dilation with sampling, we get, when the event in item 2 occurs:

1) $\kappa(G, H) \leq \kappa(G, H')\, \kappa(H', H) \leq 3 \log^2 n$;

2) the number of nontree edges is $O(m)$;

3) $\text{stretch}_{T'}(E(H) \setminus E(T)) \le m/\log n$: by Claim 23.4, each of the N samples has stretch s/N, where $s = \text{stretch}_{T'}(E(H') \setminus E(T)) = m/\log n$, so their total stretch is s.

If we do not combine samples but keep a multigraph, then in addition:

4) every nontree sample $\hat{e}$ has $\text{stretch}_{T'}(\hat{e}) = s/N \le 1/(C_s \log n \log(1/\delta))$, which is $O(1/\log n)$.

23.3 The critical trick to get a one-log solver

This is the critical trick of Lecture 22, in the notation of its sampling algorithm $\text{Sample}(G, T)$: the nontree edge e is sampled with probability $p_e = w_e \,\text{ER}_T(e)/s$, where $s = \text{stretch}_T(E(G) \setminus E(T))$, and $N = \lceil C_s \, s \log n \log(1/\delta) \rceil$ samples are drawn.

Claim 23.4. *Let $\hat{e}$ be a sampled copy of a nontree edge e in the graph returned by* $\text{Sample}(G, T)$*, with its normalized weight $w_{\hat{e}} = \frac{w_e}{N p_e}$. Then*

$$\text{stretch}_T(\hat{e}) = \frac{s}{N} \le \frac{1}{C_s \log n \log(1/\delta)}.$$

Proof. Since $p_e = w_e \,\text{ER}_T(e)/s$,

$$w_{\hat{e}} = \frac{w_e}{p_e} \cdot \frac{1}{N} = \frac{w_e \, s}{w_e \,\text{ER}_T(e)} \cdot \frac{1}{N} = \frac{s}{N \,\text{ER}_T(e)},$$

where the factor w_e cancels. Hence $\text{stretch}_T(\hat{e}) = w_{\hat{e}} \,\text{ER}_T(e) = s/N$, and $N \ge C_s \, s \log n \log(1/\delta)$ gives the bound. □

23.4 Phase II of the solver chain construction

The solver chain starts with $(G_1, T_1) = (H, T')$, where H and T' are as above, so T_1 is the spanning tree of G_1 with its weights in G_1. Along the chain, the following decrease exponentially:

1) the number of nontree edges;

2) the number of vertices, once vertices of degree one and two are eliminated (Theorem 23.5).

Fix a constant $t > 1$. To get G_2:

1) dilate the tree: let $H_1' = G_1 + (t-1)\,T_1$, whose tree is $T_2 = tT_1$;

2) sample: let $G_2 = \text{Sample}(H_1', T_2)$, that is, sample the nontree edges of H_1' using resistance estimates from T_2. The tree of G_2 is T_2.

By Claim 23.4, the nontree edges of G_1 are samples of stretch at most $1/(C_s \log n \log(1/\delta))$ with respect to T_1. The analysis of Phase 2 in Lecture 22, with $k = t$, therefore gives:

1) $|E(G_2) \setminus E(T)| \le \lceil |E(G_1) \setminus E(T)|/t \rceil$;

2) $\kappa(G_1, G_2) \le 3t$ with high probability;

3) preconditioned Chebyshev iteration $\text{Pre-Cheb}(G_1, G_2)$ needs $O(\sqrt{t})$ iterations per constant factor of error reduction (Lectures 14 and 17).

We compute G_3 from G_2 in the same way, and so on, as long as at least $\sqrt{n}$ nontree edges remain, as in Phase 2 of Lecture 22. Thus each step divides the number of nontree edges by about t: by item 1, a graph with m' nontree edges is followed by one with at most $\lceil m'/t \rceil$. With $t = 4$, the count shrinks to at most $\lceil m'/4 \rceil$, about $\frac{1}{4}$ of its value, at each step.

The vertex count, however, does not change: every G_i contains a spanning tree on all n vertices. The idea is therefore to pivot out the vertices of degree one and two between each G_i and G_{i+1}.

23.5 Removing degree one and two nodes using Gaussian elimination

Theorem 23.5. *Let G be a multigraph with $n \geq 2$ vertices, let T be a spanning tree of G, and let m be the number of nontree edges. If no vertex of G has degree one or two, counting parallel edges with multiplicity, then $n \leq 2m$.*

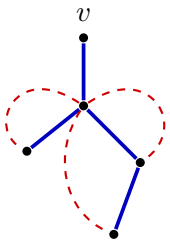
Proof. This is the theorem on degree one and two nodes in Lecture 22, and the proof is the same. Let n_1, n_2 and n_3 be the numbers of vertices of degree 1, 2 and at least 3 in T. A tree with $n \geq 2$ vertices has $n_1 \geq n_3 + 2$ leaves, so $n \leq 2n_1 + n_2$. Every vertex has degree at least 3 in G, so each leaf of T is an endpoint of at least two nontree edges and each vertex of degree 2 in T of at least one. Hence $2m \geq 2n_1 + n_2 \geq n$. □

The full algorithm pivots out vertices of degree one and two after each level of Phase 2.

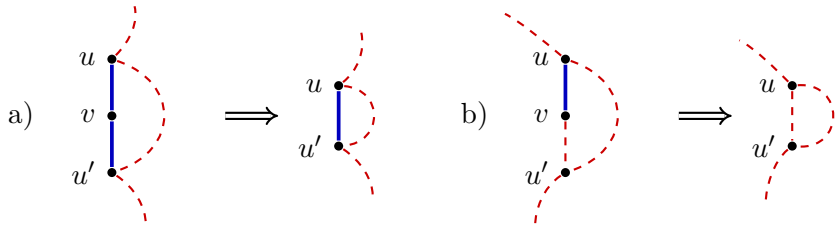
23.6 We must check that the spanning tree is low stretch

Eliminating a vertex changes the graph and its spanning tree, so we must check that the tree remains low stretch. In the figures below, tree edges are drawn solid and blue, and nontree edges dashed and red; the loose ends stand for the rest of the graph.

1) A vertex of degree one is joined to the rest of the graph by a single edge, which is a tree edge, since T spans the graph:



2) A vertex v of degree two has two edges, and at least one of them is a tree edge, since T connects v to the other vertices. There are two cases. In case a) both edges are tree edges, and eliminating v replaces the tree path u, v, u' by a single tree edge (u, u'). In case b) the edge (v, u) is a tree edge and the edge (v, u') is a nontree edge; if $u' \neq u$, as in the figure, eliminating v replaces them by a single nontree edge (u, u').



Proposition 23.6. *Let G be a multigraph on $n \geq 3$ vertices with a spanning tree T, and let v be a vertex of degree one or two in G. Eliminating v, that is, replacing L_G by its Schur complement onto $V \setminus \{v\}$, gives the Laplacian of a graph G' on $V \setminus \{v\}$ with a spanning tree T' such that*

$$\mathrm{stretch}_{T'}\big(E(G') \setminus E(T')\big) \leq \mathrm{stretch}_{T}\big(E(G) \setminus E(T)\big).$$

In cases 1 and a), the stretch of every remaining nontree edge is unchanged.

Proof. For each vertex $x \neq v$, let a_x be the total weight of the edges between v and x, let a be the vector of these weights, and let $d_v = \sum_x a_x$. With v ordered last,

$$L_G = \begin{pmatrix} L_{G-v} + \mathrm{diag}(a) & -a \\ -a^{\top} & d_v \end{pmatrix},$$

so the Schur complement onto $V \setminus \{v\}$ is $L_{G-v} + \mathrm{diag}(a) - aa^{\top}/d_v$. The matrix $\mathrm{diag}(a) - aa^{\top}/d_v$ is the Laplacian of the graph on the neighbors of v in which any two neighbors $x \neq y$ are joined by an edge of weight $a_x a_y / d_v$; it is 0 if v has a single neighbor. Hence $G' = G - v$ if v has a single neighbor. If v has two distinct neighbors u and u', joined to v by edges of weights a_u and $a_{u'}$, then G' is $G - v$ plus an edge (u, u') of weight $a_u a_{u'}/(a_u + a_{u'})$, whose resistance $1/a_u + 1/a_{u'}$ equals that of the path u, v, u'.

Case 1: v has degree one. Its edge is a tree edge and v is a leaf of T. Let $T' = T - v$. A tree path between two vertices other than v does not pass through the leaf v, so $\mathrm{ER}_{T'}(x, y) = \mathrm{ER}_T(x, y)$ for all $x, y \neq v$. The nontree edges of G' are those of G, with the same stretch.

Case a): both edges at v are tree edges. Their other endpoints u and u' are distinct, since T has no parallel edges. Let $T' = T - v + (u, u')$, a spanning tree of G'. For $x, y \neq v$, the path from x to y in T' is the path in T, with u, v, u' replaced by the edge (u, u') if it passes through v; since the two have the same resistance, $\mathrm{ER}_{T'}(x, y) = \mathrm{ER}_T(x, y)$. Again the nontree edges keep their stretch.

Case b): the edge (v, u) of weight a is a tree edge and the edge (v, u') of weight b is a nontree edge. Then v is a leaf of T; let $T' = T - v$. As in Case 1, the nontree edges of G other than (v, u') keep their stretch. If $u' = u$, the edge (v, u') disappears and there is no new edge. If $u' \neq u$, the new edge (u, u') of weight $ab/(a + b)$ is a nontree edge of G'. The tree path from v to u' is the edge (v, u) followed by the tree path from u to u', so $\mathrm{ER}_T(v, u') = 1/a + \mathrm{ER}_T(u, u')$. Hence the new edge has stretch

$$\frac{ab}{a+b}\,\mathrm{ER}_{T'}(u, u') \leq b\,\mathrm{ER}_T(u, u') \leq b\Big(\frac{1}{a} + \mathrm{ER}_T(u, u')\Big) = b\,\mathrm{ER}_T(v, u'),$$

which is the stretch of the edge (v, u') that it replaces.

In every case the total stretch of the nontree edges does not increase. □

24 Random Walks with Restarts, Local PageRank, Spilling Paint

Lecture of April 3, 2020. Based on Gary L. Miller's handwritten notes.

A set of vertices with small conductance is separated from the rest of the graph by little edge weight. This lecture asks how to find such a set near a given vertex while looking at only a small part of the graph. The tool is the random walk with restarts, whose stationary vector is the local (personal) PageRank vector. We show that this vector exists, write it as a sum of lossy random walks and as the dry paint of a paint-spilling process, and approximate it locally by pushing wet paint. We then read the stationary vector as a circulation, state the main theorem on finding sets of low conductance among sweep sets, and begin an analysis through a grounded Laplacian and the Rayleigh quotients of Lecture 12.

24.1 Personal PageRank and spilling paint

Throughout this lecture, $G = (V, E, w)$ is an undirected graph with positive edge weights w_e, with $n = |V|$ vertices and $m = |E|$ edges, in which every vertex has positive degree. Let A be its weighted adjacency matrix, $d(v) = \sum_{x \in V} A_{vx}$ the degree of v, and $D = \operatorname{diag}(d)$ the degree matrix.

Definition 24.1. For $S \subseteq V$, the *volume* and the *boundary* of S are

$$\operatorname{vol}(S) \coloneqq d(S) \coloneqq \sum_{v \in S} d(v), \qquad \partial S \coloneqq \{(v, x) \in E : v \in S,\ x \notin S\},$$

and the weight of the boundary is $w(\partial S) \coloneqq \sum_{e \in \partial S} w_e$. For $\emptyset \neq S \subsetneq V$, the *conductance* of S is

$$\Phi(S) \coloneqq \operatorname{Conductance}(S) \coloneqq \frac{w(\partial S)}{\min\{d(S), d(V \setminus S)\}}.$$

For unit weights, $w(\partial S) = |\partial S|$.

Given a vertex $v \in V$, we want to find a small set S with $v \in S$ and small conductance. The idea is to find S with local random walks, which explore only the part of the graph near v. This leads to local PageRank, the local version of Google's PageRank.

24.2 Two random walks

A *distribution* on V is a vector $p \in \mathbb{R}^n$ with $p \geq 0$ and $\|p\|_1 = 1$.

Definition 24.2. The *regular* or *eager* walk and the *lazy* walk update a distribution $p^{(t)}$ by

$$p^{(t+1)} = W p^{(t)}, \quad W \coloneqq A^\top D^{-1} = A D^{-1}, \qquad \text{and} \qquad p^{(t+1)} = \hat{W} p^{(t)}, \quad \hat{W} \coloneqq \tfrac{1}{2}(I + W),$$

respectively. Let u be a distribution on V, for example the characteristic vector χ_v of a vertex v, and let $0 < \alpha < 1$. The eager and the lazy *walk with resets* to u are

$$\begin{aligned} p^{(t+1)} &= \alpha u + (1 - \alpha) W p^{(t)} && \text{(eager)}, \\ p^{(t+1)} &= \alpha u + (1 - \alpha) \hat{W} p^{(t)} && \text{(lazy)}. \end{aligned}$$

At each step, the walk with resets restarts from u with probability α and otherwise takes a step of the eager or the lazy walk. To show that it has a stationary vector, we bound eigenvalues. For $M \in \mathbb{R}^{n\times n}$, let $\lambda(M)$ be the set of eigenvalues of M, and let

$$|\lambda(M)| \coloneqq \max\{|\lambda| : \lambda \in \lambda(M)\}$$

be its spectral radius. In general, the eigenvalues and eigenvectors of M are complex, and $|a+ib| = \sqrt{a^2+b^2}$.

Lemma 24.3. *Let $M \in \mathbb{R}^{n\times n}$ with $M \geq 0$ entrywise, and let s be the maximum row sum of M. Then $|\lambda(M)| \leq s$.*

Proof. Let $\lambda \in \lambda(M)$, and let $x \in \mathbb{C}^n$ be nonzero with $Mx = \lambda x$. Fix an index i with $|x_i| = \max_j |x_j|$, so $|x_i| > 0$, and let s_i be the sum of row i of M. If $s_i = 0$, then row i of M is zero, so $\lambda x_i = (Mx)_i = 0$ and $\lambda = 0 \leq s$. If $s_i > 0$, then

$$\frac{\lambda}{s_i}\, x_i = \Big(\frac{1}{s_i} Mx\Big)_i = \sum_j \frac{M_{ij}}{s_i}\, x_j$$

is a convex combination of $x_1, \ldots, x_n$, so its absolute value is at most $|x_i|$. Hence $|\lambda/s_i| \leq 1$, that is, $|\lambda| \leq s_i \leq s$. □

Theorem 24.4. *Let u be a distribution on V and $0 < \alpha < 1$. There is a unique vector $P_u \in \mathbb{R}^n$ with*

$$P_u = \alpha u + (1-\alpha) W P_u, \tag{$*$}$$

namely $P_u = \alpha\big(I - (1-\alpha)W\big)^{-1} u$.

Proof. Equation ($*$) is equivalent to $\big[I - (1-\alpha)W\big]P_u = \alpha u$, so it suffices to show that $I-(1-\alpha)W$ is nonsingular, that is, that none of its eigenvalues is 0.

The matrix $W = AD^{-1}$ is nonnegative, and each column of W sums to 1, since $\sum_x A_{xv}/d(v) = 1$. Its transpose $D^{-1}A$ is nonnegative with row sums 1, and it has the same eigenvalues as W. Applying Lemma 24.3 to $D^{-1}A$ gives $|\lambda(W)| \leq 1$.

If $Wx = \lambda x$, then $\big(I - (1-\alpha)W\big)x = \big(1 - (1-\alpha)\lambda\big)x$, and conversely; so the eigenvalues of $I - (1-\alpha)W$ are the numbers $\mu = 1 - (1-\alpha)\lambda$ with $\lambda \in \lambda(W)$. Every such μ satisfies

$$|\mu| \geq 1 - (1-\alpha)|\lambda| \geq 1 - (1-\alpha) = \alpha > 0. \tag{24.1}$$

Hence $I - (1-\alpha)W$ is nonsingular, and $P_u = \big(I - (1-\alpha)W\big)^{-1}\alpha u$ is the unique solution. □

We call P_u the *stationary* vector of the walk with resets to u; it is the local (personal) PageRank vector of u, and we abbreviate local PageRank as LPR.

24.3 Spectral theorem

Fact 24.5 (Spectral theorem)**.** *If $M \in \mathbb{R}^{n\times n}$ is symmetric, then $M = U^\top \Lambda U$, where $U^\top U = I$ and Λ is a real diagonal matrix.*

Consequently, if M is symmetric and $|\lambda(M)| < 1$, then

$$(I - M)^{-1} = I + M + M^2 + \cdots,$$

because $M^t = U^\top \Lambda^t U$ and $\sum_{t \geq 0} \lambda^t = 1/(1-\lambda)$ for each eigenvalue λ of M. The matrix $W = AD^{-1}$ is not symmetric, but still

$$P_u = \alpha \sum_{t \geq 0} (1-\alpha)^t W^t u. \tag{24.2}$$

To see this, write $W = D^{1/2} N D^{-1/2}$ with $N = D^{-1/2} A D^{-1/2}$ symmetric. Then N and W have the same eigenvalues, so $|\lambda((1-\alpha)N)| \leq 1 - \alpha < 1$, and

$$\big(I - (1-\alpha)W\big)^{-1} = D^{1/2}\big(I - (1-\alpha)N\big)^{-1} D^{-1/2} = \sum_{t \geq 0} (1-\alpha)^t D^{1/2} N^t D^{-1/2} = \sum_{t \geq 0} (1-\alpha)^t W^t.$$

Thus P_u is a sum of lossy random walks from u: the eager walk from u loses a fraction α of its remaining mass at each step, and $\alpha(1-\alpha)^t W^t u$ is the mass it loses at step t. In particular, $P_u \geq 0$.

24.4 Spilling paint view of LPR

Start with a bucket of paint at a vertex v, or more generally with wet paint distributed as u. At each time t, a fraction α of the wet paint spills and dries where it is, and the remaining wet paint moves to a random neighbor, as in the eager walk. Let $s^{(t)}$ and $r^{(t)}$ be the distributions of dry and of wet paint at time t, with $s^{(0)} = 0$ and $r^{(0)} = u$. The evolution equations are

$$s^{(t+1)} = s^{(t)} + \alpha r^{(t)}, \qquad r^{(t+1)} = (1-\alpha) W r^{(t)}, \quad \text{so} \quad r^{(t)} = (1-\alpha)^t W^t r^{(0)}.$$

The dry paint therefore converges to

$$s^\infty = \alpha \sum_{t \geq 0} r^{(t)} = \alpha \sum_{t \geq 0} (1-\alpha)^t W^t u = P_u,$$

by (24.2): the stationary vector is the final distribution of the dry paint. We write $s^\infty(\text{eager}, \alpha) = P_u$, and $s^\infty(\text{lazy}, \alpha)$ for the same quantity with $\hat{W}$ in place of W. Since $\hat{W} \geq 0$ also has column sums 1, the proof of Theorem 24.4 applies to $\hat{W}$, and $s^\infty(\text{lazy}, \alpha) = \alpha\big(I - (1-\alpha)\hat{W}\big)^{-1} u$.

Claim 24.6. *Let u be a distribution on V, let $0 < \alpha < 1$, and set $\beta = 2\alpha/(1+\alpha)$. Then $s^\infty(\text{lazy}, \alpha) = s^\infty(\text{eager}, \beta)$. Since $\alpha < \beta < 2\alpha$, the regular and the lazy walk give the same local PageRank vectors, up to a constant factor in the reset probability.*

Proof. Since $\hat{W} = I/2 + W/2$,

$$\begin{aligned} s^\infty(\text{lazy}, \alpha) &= \alpha\Big(I - (1-\alpha)\Big(\frac{I}{2} + \frac{W}{2}\Big)\Big)^{-1} u = \alpha\Big(\frac{1+\alpha}{2} I - \frac{1-\alpha}{2} W\Big)^{-1} u \\ &= \frac{2\alpha}{1+\alpha}\Big(I - \frac{1-\alpha}{1+\alpha} W\Big)^{-1} u. \end{aligned}$$

With $\beta = 2\alpha/(1+\alpha)$ we have $1 - \beta = (1-\alpha)/(1+\alpha)$, so

$$s^\infty(\text{lazy}, \alpha) = \beta\big(I - (1-\beta)W\big)^{-1} u = s^\infty(\text{eager}, \beta). \qquad \square$$

24.5 Pushing wet paint

Suppose that dry paint $s \geq 0$ and wet paint $r \geq 0$ are distributed on V. Running the paint-spilling process from (s, r) ends with the dry paint

$$P_{s,r} = s + \alpha \sum_{t \geq 0} (1-\alpha)^t W^t r = s + \alpha (I - (1-\alpha)W)^{-1} r.$$

A partial update moves only the wet paint at a single vertex. In the rest of this subsection, G has unit weights, so $d(v)$ is the number of neighbors of v; let $N(v)$ be the set of neighbors of v.

Definition 24.7 (Unweighted case)**.** For $v \in V$, the *update* $\mathrm{Update}_v(s, r) = (s', r')$ is

$$s'(x) = \begin{cases} s(x) + \alpha r(x) & \text{if } x = v, \\ s(x) & \text{if } x \neq v, \end{cases} \qquad r'(x) = \begin{cases} 0 & \text{if } x = v, \\ r(x) + \dfrac{1-\alpha}{d(v)} r(v) & \text{if } x \in N(v), \\ r(x) & \text{otherwise.} \end{cases}$$

An update dries the fraction α of the wet paint at v and moves the rest as one step of the eager walk.

Lemma 24.8. *For all $s, r \geq 0$ and $v \in V$, if $(s', r') = \mathrm{Update}_v(s, r)$, then $P_{s',r'} = P_{s,r}$.*

Proof. This lemma is from Spielman's notes [Spi10]. Let $R = (I - (1-\alpha)W)^{-1}$. For unit weights, $W\chi_v = \frac{1}{d(v)} \sum_{x \in N(v)} \chi_x$, so

$$s' = s + \alpha r(v) \chi_v, \qquad r' = r - r(v)\chi_v + (1-\alpha) r(v) W \chi_v = r - r(v)(I - (1-\alpha)W)\chi_v.$$

Hence

$$P_{s',r'} = s + \alpha r(v)\chi_v + \alpha R r - \alpha r(v) R (I - (1-\alpha)W)\chi_v = P_{s,r} + \alpha r(v)\chi_v - \alpha r(v)\chi_v = P_{s,r}.$$

□

Algorithm 24.9 (ApproxPainting(u, α, ε))**.**

Input: a vertex u, a reset probability $0 < \alpha < 1$ and a tolerance $\varepsilon > 0$.

Init: $s = 0$ and $r = \chi_u$.

Loop: while there is $v \in V$ with $r(v) \geq \varepsilon d(v)$, choose $v \in \arg\max_{x \in V} r(x)/d(x)$ and replace (s, r) by $\mathrm{Update}_v(s, r)$.

Output: s and r.

Theorem 24.10. ApproxPainting(u, α, ε) *terminates after at most $1/(\varepsilon\alpha)$ iterations of the loop.*

Proof. An update keeps $s, r \geq 0$ and preserves the total paint $\|s\|_1 + \|r\|_1$, which is 1 at the start: it moves $\alpha r(v)$ to dry paint and $(1-\alpha)r(v)$ in equal shares to the $d(v)$ neighbors of v. Each iteration updates a vertex v with $r(v) \geq \varepsilon d(v) \geq \varepsilon$, so it increases $\|s\|_1$ by $\alpha r(v) \geq \alpha\varepsilon$. Since $\|s\|_1 \leq 1$ throughout, there are at most $1/(\varepsilon\alpha)$ iterations. □

24.6 Understanding the stationary P_v for walks with resets from v

From now on, the reset distribution is a single vertex $u \in V$; we identify u with χ_u and write P_u for its stationary vector.

Definition 24.11. Let $g_u(v) \coloneqq P_u(v)/d(v)$ for $v \in V$. Order the vertices as $v_1, \dots, v_n$ so that $g_u(v_1) \geq g_u(v_2) \geq \cdots \geq g_u(v_n)$, with ties broken arbitrarily, and let $S_k \coloneqq \{v_1, \dots, v_k\}$ for $1 \leq k \leq n$. The sets S_k are the *sweep sets* of P_u.

We write $P_i = P_u(v_i)$ and $g(i) = g_u(v_i)$ when u is fixed. The stationary P_u carries two kinds of flow:

1) the *reset-flow*, which carries probability back to u;

2) the *graph-flow*, which moves probability along the edges of G.

Together they form a circulation (Lemma 24.12 below).

24.7 The stationary as a flow (circulation)

Consider one step of the walk with resets started from P_u. The mass $(1-\alpha)P_i$ leaves v_i along edges, a fraction $A_{ij}/d(i)$ of it along each edge (i,j), so $(1-\alpha)A_{ij}\, g(i)$ moves from v_i to v_j. Thus, up to the factor $1-\alpha$, $g(i)$ is the probability of leaving v_i on a fixed edge of unit weight out of v_i. The net probability that moves along the edge $(i,j) \in E$ from v_i to v_j is

$$f_{ij} = (1-\alpha)A_{ij}\big[g(i) - g(j)\big],$$

which for unit weights is $(1-\alpha)\big[g(i) - g(j)\big]$. We view f as a graph-flow on the edges. The reset-flow moves the mass αP_i from each v_i to u.

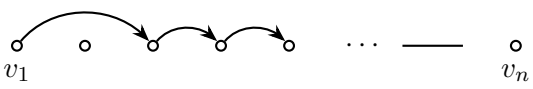
Lemma 24.12. *Let $u \in V$. Then $\mathbf{1}^\top P_u = 1$. The graph-flow and the reset-flow form a circulation: for every vertex $v_i \neq u$, the net graph-flow into v_i equals the reset-flow αP_i out of v_i, and the net graph-flow out of u equals the total reset-flow $\alpha \sum_{v_j \neq u} P_j$ into u.*

Proof. Since the columns of W sum to 1, multiplying $(*)$ by $\mathbf{1}^\top$ gives $\mathbf{1}^\top P_u = \alpha + (1-\alpha)\mathbf{1}^\top P_u$, so $\mathbf{1}^\top P_u = 1$. Since $Dg = P_u$ and $Ag = AD^{-1}P_u = WP_u$, the net graph-flow out of v_i is

$$\sum_j (1-\alpha)A_{ij}\big[g(i) - g(j)\big] = (1-\alpha)\big[d(i)g(i) - (Ag)_i\big] = (1-\alpha)\big[P_i - (WP_u)_i\big] = \alpha\chi_u(v_i) - \alpha P_i,$$

where the last step uses $(1-\alpha)(WP_u)_i = P_i - \alpha\chi_u(v_i)$, by $(*)$. For $v_i \neq u$ this is $-\alpha P_i$. For $v_i = u$ it is $\alpha(1 - P_u(u)) = \alpha \sum_{v_j \neq u} P_j$, since $\mathbf{1}^\top P_u = 1$. □

Since $g(i) \geq g(j)$ for $i \leq j$, every edge carries its graph-flow from left to right in the order $v_1, \dots, v_n$, that is, $f_{ij} \geq 0$ for $i < j$:

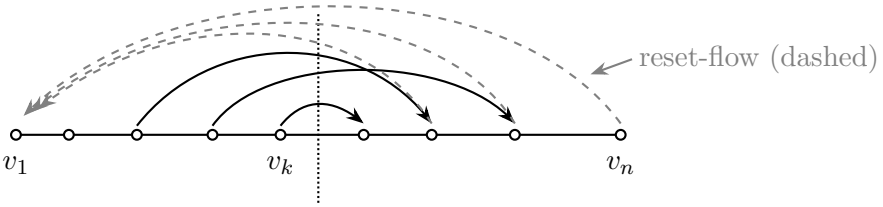


Since graph-flow and reset-flow together form a circulation, the reset-flow should run from right to left. The next claim makes this precise.

Claim 24.13. *For every $v \neq u$ we have $g_u(u) > g_u(v)$. Hence v_1 is the reset vertex u.*

Proof. Let v maximize g_u over V. Every edge at v carries its graph-flow out of v, or none, so the net graph-flow into v is at most 0. Suppose that $v \neq u$. By Lemma 24.12, the net graph-flow into v equals $\alpha P_u(v)$. Since $P_u \geq 0$ and $W \geq 0$, $(*)$ gives $P_u \geq \alpha\chi_u$, so $P_u(v) = d(v)g_u(v) \geq d(v)g_u(u) \geq \alpha d(v)/d(u) > 0$, a contradiction. Thus every vertex at which g_u is maximal is the reset vertex, and $v_1 = u$. □

In particular, $P_1 = P_u(u) \geq \alpha$. For $1 \leq k < n$, let f_k be the graph-flow crossing from S_k to $V \setminus S_k$, just after v_k in the order:



Every edge from S_k to $V \setminus S_k$ carries flow out of S_k, and reset-flow enters S_k exactly from $v_{k+1}, \ldots, v_n$, since $u = v_1 \in S_k$. Summing Lemma 24.12 over the vertices of S_k therefore gives

$$f_k = \text{reset-flow from } v_{k+1}, \ldots, v_n \text{ to } v_1 = \alpha(P_{k+1} + \cdots + P_n). \tag{24.3}$$

24.8 Main theorem

In words: if $C \subseteq V$ has low conductance and we run LPR from a random vertex $u \in C$, chosen by degree, with a reset probability $\alpha \geq 20\,\Phi(C)$, then with probability at least $1/2$ one of the sweep sets S_k has low conductance, namely $O(\sqrt{\alpha \log d(C)})$.

Theorem 24.14 (Andersen–Chung–Lang [ACL07, Theorem 5.3])**.** *Suppose that G has unit weights. Let $\emptyset \neq C \subseteq V$ with $d(C) \leq d(V)/2$ and $\Phi(C) \leq \phi$, where $0 < \phi < 1/20$, and let $\alpha = 20\phi$. Choose $u \in C$ at random with $\Pr[u = v] = d(v)/d(C)$ for $v \in C$, and let $S_1, \ldots, S_n$ be the sweep sets of P_u. Then with probability at least $1/2$ there is a $k < n$ such that*

$$\Phi(S_k) < \sqrt{30\alpha \ln(10\sqrt{d(C)})} = \sqrt{600\phi \ln(10\sqrt{d(C)})}.$$

In particular, $\Phi(S_k) = O(\sqrt{\phi \log d(C)})$, and since $d(C) \leq d(V)/2 = m$, $\Phi(S_k) = O(\sqrt{\phi \log m})$, with absolute constants in the $O(\cdot)$.

Proof. We derive the theorem from two results of Andersen, Chung and Lang, which we state without proof; the derivation follows their proof of [ACL07, Theorem 5.3]. They work with graphs with unit weights and with the lazy walk. For a teleport probability $0 < \alpha' \leq 1$, their PageRank vector of u is the unique solution $p = \mathrm{pr}_{\alpha'}(\chi_u)$ of $p = \alpha'\chi_u + (1 - \alpha')\hat{W}p$ (they write vectors as rows), that is, $\mathrm{pr}_{\alpha'}(\chi_u) = s^{\infty}(\text{lazy}, \alpha')$ with reset distribution χ_u. Their sweep sets of p are the sets $\{v_1, \ldots, v_j\}$ for an ordering $v_1, \ldots, v_N$ of the N vertices with $p(v) > 0$ in which $p(v)/d(v)$ does not increase. The two results are the following.

(a) [ACL07, Theorem 5.1] For every $\emptyset \neq C \subsetneq V$ and $0 < \alpha' \leq 1$ there is a set $C' \subseteq C$ with $d(C') \geq d(C)/2$ such that $\mathrm{pr}_{\alpha'}(\chi_v)(C) \geq 1 - \Phi(C)/\alpha'$ for every $v \in C'$.

(b) [ACL07, Theorem 4.1] If $v \in V$, $S \subseteq V$ and $\delta > 0$ satisfy $\mathrm{pr}_{\alpha'}(\chi_v)(S) - d(S)/d(V) > \delta$, then some sweep set $S' \neq V$ of $\mathrm{pr}_{\alpha'}(\chi_v)$ has

$$\Phi(S') < \sqrt{\frac{12\alpha' \ln\bigl(4\sqrt{d(S)}/\delta\bigr)}{\delta}}.$$

They write log; their proof goes through with the natural logarithm, since $(1-x)^t \le e^{-tx}$, and since $\ln(4\sqrt{d(S)}/\delta) > \ln 4 > 1$, which their rounding of the number of steps to an integer needs (here $\delta < 1$ and $d(S) \ge 1$). They state the bound for the least conductance of a sweep set, and their proof (of their Theorem 4.5) never uses a conductance of the sweep set V, so the set it finds is not V.

Let $\alpha' = \alpha/(2-\alpha)$. Then $2\alpha'/(1+\alpha') = \alpha$, so $P_u = s^\infty(\text{eager}, \alpha) = s^\infty(\text{lazy}, \alpha') = \mathrm{pr}_{\alpha'}(\chi_u)$ by Claim 24.6. The vertices with $P_u(v) > 0$ come first in the order of Definition 24.11, so the sweep sets of $\mathrm{pr}_{\alpha'}(\chi_u)$ in their sense are among $S_1, \dots, S_n$. Since $0 < \alpha < 1$, we have $\alpha/2 < \alpha' < \alpha$, and therefore $\Phi(C) \le \phi = \alpha/20 < \alpha'/10$. This is where the reset probability must be a large multiple of $\Phi(C)$: it makes most of the paint dry inside C.

Let C' be the set of (a) for C and α'. Then $\Pr[u \in C'] = d(C')/d(C) \ge 1/2$. Fix $u \in C'$. By (a), $P_u(C) \ge 1 - \Phi(C)/\alpha' > 9/10$, and $d(C)/d(V) \le 1/2$, so $P_u(C) - d(C)/d(V) > 2/5$. By (b) with $v = u$, $S = C$ and $\delta = 2/5$, some $S_k \neq V$, that is, some $k < n$, has

$$\Phi(S_k) < \sqrt{30\alpha' \ln\bigl(10\sqrt{d(C)}\bigr)} < \sqrt{30\alpha \ln\bigl(10\sqrt{d(C)}\bigr)} = \sqrt{600\phi \ln\bigl(10\sqrt{d(C)}\bigr)}.$$

Finally, $d(C) \ge 2$. If no edge leaves C, then a vertex of C and a neighbor of it, two distinct vertices of degree at least 1, both lie in C. Otherwise $w(\partial C) \ge 1$, and since $d(C) \le d(V \setminus C)$, we have $d(C) = w(\partial C)/\Phi(C) \ge 1/\phi > 20$. Hence $\ln\bigl(10\sqrt{d(C)}\bigr) = \ln 10 + \frac{1}{2}\ln d(C) \le 4 \ln d(C)$, because $\ln 10 < 3.4 \ln 2$, and $\Phi(S_k) < \sqrt{2400\,\phi \ln d(C)}$. Since $d(C) \le d(V)/2 = m$, also $\Phi(S_k) < \sqrt{2400\,\phi \ln m}$. □

Remark 24.15. Up to the constant, the bound of Theorem 24.14 also holds for the approximation computed by ApproxPainting (Algorithm 24.9). Let G, C, ϕ, α and u be as in Theorem 24.14, let $\alpha' = \alpha/(2-\alpha)$ as in the proof, let $\varepsilon = 1/(10\,d(C))$, and let (s, r) be the output of ApproxPainting(u, α, ε). By Lemma 24.8, $s + \alpha(I - (1-\alpha)W)^{-1} r = P_u$, so $s = \alpha(I - (1-\alpha)W)^{-1}(\chi_u - r)$, and the loop has stopped, so $0 \le r(v) < \varepsilon\, d(v)$ for every v. The computation in the proof of Claim 24.6 holds for every vector in place of u, so $s = \alpha'(I - (1-\alpha')\hat{W})^{-1}(\chi_u - r)$: in the words of Andersen, Chung and Lang, s is an ε-approximate PageRank vector for $\mathrm{pr}_{\alpha'}(\chi_u)$. Their Theorem 5.3 [ACL07] assumes $\Phi(C) \le \alpha'/10$, $d(C) \le \frac{2}{3}d(V)$, $\varepsilon \le 1/(10\,d(C))$ and $u \in C'$, which hold with probability at least $1/2$, and then gives a sweep set of s other than V, ordered by $s(v)/d(v)$ as in Definition 24.11, with conductance $O\bigl(\sqrt{\alpha' \log d(C)}\bigr) = O\bigl(\sqrt{\phi \log d(C)}\bigr)$. By Theorem 24.10, ApproxPainting performs at most $1/(\varepsilon\alpha) = d(C)/(2\phi)$ updates, a number that does not depend on n.

Remark 24.16. Theorem 24.14 bounds the conductance of S_k, not how much of S_k lies inside C. About C, its proof gives a statement on PageRank mass: $P_u(C) > 9/10$ for every $u \in C'$. A bound on volume is known for a different rule for choosing the sweep set. Andersen and Chung [AC07, Theorem 3] work with unit weights and the eager walk. They fix a target conductance $0 < \theta < 1/3$ and set $\alpha = \theta^2/(8H(2m))$, where $H(2m) = \sum_{j=1}^{2m} 1/j$. Their algorithm Local Partition computes an approximation p of P_u and returns a sweep set of p at which the

ratios $p(v)/d(v)$ drop sharply. Suppose that $d(C) \le d(V)/2$, that $\Phi(C) \le \alpha/(80H(2m))$, that u lies in a certain subset of C that holds at least half of $d(C)$, and that the algorithm is given the target volume $d(V)$. Then it returns a set S with $\Phi(S) \le 3\theta$ and $d(S \cap C) \ge \frac{9}{10}\, d(S)$. For the same α, this hypothesis on $\Phi(C)$ is stronger than the hypothesis $\Phi(C) \le \alpha/20$ of Theorem 24.14 by the factor $4H(2m)$.

24.9 Analyzing LPR using Cheeger's inequality

We try to analyze LPR with the tools of Cheeger's inequality (Lecture 12). Assume that we know the stationary vector $P = P_u$ exactly. By Theorem 24.4, P is the unique vector with

$$(I - (1-\alpha)W)P = \alpha u.$$

We want the values $g_i = P_i/d_i$, that is, $g = D^{-1}P$, or $P = Dg$. Since $WD = A$, the vector g is the unique solution of

$$(D - (1-\alpha)A)g = \alpha u.$$

Let $L = D - A$. Then $D - (1-\alpha)A = \alpha D + (1-\alpha)L$, so

$$(\alpha D + (1-\alpha)L)g = \alpha u, \qquad \Big(\frac{\alpha}{1-\alpha}\, D + L\Big)g = \frac{\alpha}{1-\alpha}\, u.$$

With $\beta = \alpha/(1-\alpha)$, this reads

$$(\beta D + L)g = \beta u.$$

We can view βD as wires to a ground node.

Definition 24.17. Let G_β be the graph G together with a ground vertex v_{n+1} that is joined to each $v_i \in V$ by an edge of weight βd_i. Its Laplacian $\bar{L} = \bar{L}_\beta$ is $L + \beta D$ with an added row and column:

$$\bar{L}_\beta = \begin{pmatrix} L + \beta D & -\beta d \\ -\beta d^{\top} & \beta d(V) \end{pmatrix},$$

where $d = D\mathbf{1}$ is the vector of degrees.

Let $\bar{g} = (g; 0)$ and $\bar{u} = (u; -1)$ in $\mathbb{R}^{n+1}$. Then

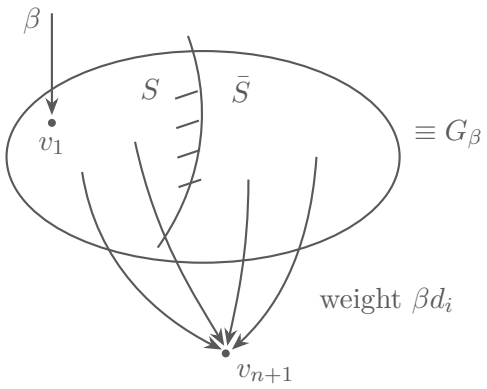
$$\bar{L}\bar{g} = \beta\bar{u}, \qquad \bar{g}_{n+1} = 0 :$$

the first n rows are $(L + \beta D)g = \beta u$, and the last row is $-\beta d^{\top} g = -\beta \mathbf{1}^{\top} P = -\beta$, because $\mathbf{1}^{\top} P = 1$ by Lemma 24.12.

We are only interested in sweep cuts, which depend only on the order of the values of g, so we can let $\bar{g}_{n+1}$ float. The graph G_β is connected, since every v_i is joined to v_{n+1}, so by Theorem 2.13 the solutions of $\bar{L}x = \beta\bar{u}$ are the vectors $\bar{g} + c\mathbf{1}$ with $c \in \mathbb{R}$, and all of them give the same sweep sets. Since $\bar{u} \perp \mathbf{1}$, we may therefore take

$$\bar{g} = \beta \bar{L}^{\dagger}\bar{u} = \beta \bar{L}^{\dagger} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix},$$

where the second form uses $u = v_1$ (Claim 24.13). In electrical terms, a current β enters G_β at v_1 and leaves at the ground v_{n+1}:



Next we consider a Rayleigh quotient. Fix $\emptyset \neq S \subsetneq V$, let $\bar{S} = V \setminus S$, and define $f \in \mathbb{R}^{n+1}$ by

$$f(x) = \begin{cases} a & \text{if } x \in S, \\ -b & \text{if } x \in \bar{S}, \\ 0 & \text{if } x = v_{n+1}, \end{cases} \qquad a = \operatorname{vol}(\bar{S}), \quad b = \operatorname{vol}(S).$$

We also write D for the degree matrix of G extended by 0 at v_{n+1}. When S plays the role of C in Theorem 24.14, we have $\operatorname{vol}(S) \le \operatorname{vol}(\bar{S})$ and $w(\partial S)/\operatorname{vol}(S) \le \phi$.

Claim 24.18. *If* $\operatorname{vol}(S) \le \operatorname{vol}(\bar{S})$ *and* $w(\partial S)/\operatorname{vol}(S) \le \phi$*, then*

$$\frac{f^\top \bar{L} f}{f^\top D f} = \frac{w(\partial S)\operatorname{vol}(V)}{\operatorname{vol}(S)\operatorname{vol}(\bar{S})} + \beta \le 2\phi + \beta.$$

Proof. The edges of G_β are the edges of G and the wires to v_{n+1}. An edge of G inside S or inside $\bar{S}$ contributes 0 to the quadratic form $f^\top \bar{L} f$ (Lemma 2.11), an edge $e \in \partial S$ contributes $w_e(a+b)^2 = w_e \operatorname{vol}(V)^2$, and the wire at v_i contributes $\beta d_i f(v_i)^2$. Hence

$$\begin{aligned} f^\top \bar{L} f &= w(\partial S)\operatorname{vol}(V)^2 + \beta\big(\operatorname{vol}(S)\,a^2 + \operatorname{vol}(\bar{S})\,b^2\big) \\ &= w(\partial S)\operatorname{vol}(V)^2 + \beta \operatorname{vol}(S)\operatorname{vol}(\bar{S})\operatorname{vol}(V), \\ f^\top D f &= \operatorname{vol}(S)\,a^2 + \operatorname{vol}(\bar{S})\,b^2 = \operatorname{vol}(S)\operatorname{vol}(\bar{S})\operatorname{vol}(V). \end{aligned}$$

Dividing gives the equality. Since $\operatorname{vol}(S) \le \operatorname{vol}(\bar{S})$, we have $\operatorname{vol}(V) \le 2\operatorname{vol}(\bar{S})$, so the first term is at most $2w(\partial S)/\operatorname{vol}(S) \le 2\phi$. □

Remark 24.19. If $\alpha \le 1/2$, then $\beta = \alpha/(1-\alpha) \le 2\alpha$, so the bound $2\phi + \beta$ of Claim 24.18 is $O(\phi)$ whenever $\alpha = O(\phi)$.

25 Solving Symmetric Diagonally Dominant (SDD) Systems

Lecture of April 8, 2020. Based on Gary L. Miller's handwritten notes.

A symmetric diagonally dominant matrix differs from a Laplacian in that its off-diagonal entries may be positive. This lecture reduces solving such systems to solving Laplacian systems. We view an SDD matrix as the Laplacian of a graph whose edges may have negative weights, and show that a change of variables makes every weight positive exactly when every cycle has positive sign. When some cycle is negative, we solve the system through the Laplacian of a double cover of the graph, and we compare the eigenvalues and eigenvectors of the two Laplacians.

25.1 Solving symmetric diagonally dominant systems

Definition 25.1. A matrix $A \in \mathbb{R}^{n\times n}$ is *symmetric diagonally dominant (SDD)* if $A^\top = A$ and $a_{ii} \geq \sum_{j\neq i} |a_{ij}|$ for every i.

Our goal is to reduce solving SDD systems to solving Laplacian systems. The first idea is a *change of variables*.

Example 25.2. The matrix $A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$ is SDD but not a Laplacian. Consider the system

$$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}.$$

We change variables to $x' = -x$ and $a' = -a$. The system becomes

$$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} -x' \\ y \end{pmatrix} = \begin{pmatrix} -a' \\ b \end{pmatrix}, \quad \text{that is,} \quad \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x' \\ y \end{pmatrix} = \begin{pmatrix} a' \\ b \end{pmatrix},$$

which is the Laplacian system

$$\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x' \\ y \end{pmatrix} = \begin{pmatrix} a' \\ b \end{pmatrix}.$$

For $i \neq j$, let $\chi_{ij} = \chi_i - \chi_j$, and let $H_{ij} = \chi_{ij}\chi_{ij}^\top$ and $P_{ij} = (\chi_i + \chi_j)(\chi_i + \chi_j)^\top$. In rows and columns i and j, these matrices are $\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$ and $\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$, and all their other entries are 0. Recall that a Laplacian A satisfies $A = \sum_{i<j}(-a_{ij})H_{ij}$. An SDD matrix A satisfies

$$A = \sum_{\substack{i<j \\ a_{ij}<0}} (-a_{ij})H_{ij} + \sum_{\substack{i<j \\ a_{ij}>0}} a_{ij}P_{ij} + \Delta,$$

where Δ is diagonal with $\Delta_{ii} = a_{ii} - \sum_{j\neq i}|a_{ij}| \geq 0$.

Definition 25.3. Let A be SDD. Its graph is $\mathrm{Graph}(A) = (V, E, W)$, where $V = \{v_1, \ldots, v_n\}$, E contains the edge v_iv_j for each pair $i \neq j$ with $a_{ij} \neq 0$, and this edge has weight $W_{ij} = -a_{ij}$. Conversely, for a graph $G = (V, E, W)$ with nonzero real edge weights, its *Laplacian* is

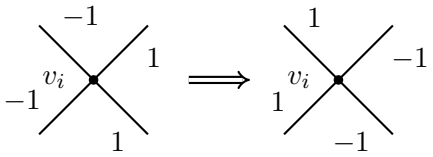
$$L_G \coloneqq \sum_{W_{ij}>0} W_{ij}H_{ij} + \sum_{W_{ij}<0} |W_{ij}|\, P_{ij},$$

the sums running over the edges v_iv_j of G.

Thus an SDD matrix is $A = L_{\mathrm{Graph}(A)} + \Delta$: we view it as the Laplacian of a graph with both positive and negative edge weights, plus a nonnegative diagonal. If all edge weights are positive, then $L_{\mathrm{Graph}(A)}$ is an ordinary Laplacian. We ask when a change of variables makes all edge weights positive. Consider changes of variables of the form

$$x'_i = -x_i \quad \text{and} \quad b'_i = -b_i. \tag{$*$}$$

In matrix form, ($*$) replaces $Ax = b$ by $(SAS)(Sx) = Sb$, where S is diagonal with $S_{ii} = -1$ and $S_{kk} = 1$ for $k \neq i$. Conjugating by S negates the off-diagonal entries in row and column i and nothing else, so ($*$) flips the sign of every edge at v_i. For example:



A *change of variables* (COV) applies ($*$) at every vertex of a set $F \subseteq V$. It replaces A by SAS, where $S = \mathrm{diag}(s)$ with $s_k = -1$ for $v_k \in F$ and $s_k = 1$ otherwise, and it changes the weight of an edge $v_y v_z$ to $s_y s_z W_{yz}$.

Definition 25.4. The *sign* of an edge e is $\mathrm{sign}(e) \in \{1, -1\}$, the sign of its weight. For a path, a cycle or a walk C, let $\mathrm{sign}(C) = \prod_{e \in C} \mathrm{sign}(e)$, where an edge traversed several times appears with its multiplicity. We call C *positive* if $\mathrm{sign}(C) = 1$ and *negative* if $\mathrm{sign}(C) = -1$.

Theorem 25.5. *Let G be a graph with nonzero real edge weights. All cycles of G have positive sign if and only if there is a change of variables after which all edge weights are positive.*

Proof. ($\Leftarrow$) A change of variables does not change the sign of a cycle: a cycle through v_i contains exactly two edges at v_i, and ($*$) flips both. If all weights are positive after the change, then every cycle is positive afterward, and hence before.

($\Rightarrow$) We may assume that G is connected, since a change of variables on one connected component does not affect the others. Fix a vertex $x \in V$.

Claim. *All walks from x to a vertex y have the same sign.*

First, every closed walk Z is positive. We induct on the length ℓ of Z. If $\ell = 0$, then $\mathrm{sign}(Z) = 1$. If no vertex repeats on Z apart from its endpoints, then either Z is a cycle, which is positive by hypothesis, or $\ell = 2$ and Z traverses one edge twice, so $\mathrm{sign}(Z) = \mathrm{sign}(e)^2 = 1$. Otherwise, some vertex occurs at two positions $i < j$ of Z that are not its two ends; cutting Z at these positions writes it as two shorter closed walks whose signs multiply to $\mathrm{sign}(Z)$, and both are positive by induction.

Let P_{xy} and P'_{xy} be walks from x to y, and let P'_{yx} be the reverse of P'_{xy}. Then $P_{xy} * P'_{yx}$ is a closed walk, so

$$1 = \mathrm{sign}(P_{xy} * P'_{yx}) = \mathrm{sign}(P_{xy}) \cdot \mathrm{sign}(P'_{yx}) = \mathrm{sign}(P_{xy}) \cdot \mathrm{sign}(P'_{xy}),$$

and hence $\mathrm{sign}(P_{xy}) = \mathrm{sign}(P'_{xy})$. This proves the claim.

Since G is connected, we may define $P(y) = \mathrm{sign}(P_{xy})$ for any walk P_{xy} from x to y. Now apply the COV ($*$) at every y with $P(y) = -1$. Let $e = v_y v_z$ be an edge. Then $P(z) = P(y)\,\mathrm{sign}(e)$, by following P_{xy} with e. Thus after the COV the sign of e is $P(y)\,\mathrm{sign}(e)P(z) = P(y)^2\,\mathrm{sign}(e)^2 = +1$. □

Algorithm 25.6 (Solving $L_G x = b$)**.** Let G be a graph with nonzero real edge weights, and $A = L_G$.

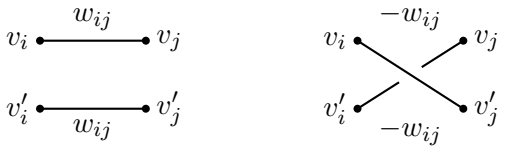
1) If all cycles of G are positive:
   a) find a change of variables S such that the graph G_S of SL_GS has positive weights (Theorem 25.5);
   b) solve the Laplacian system $L_{G_S}\tilde{x} = Sb$;
   c) undo the change of variables: return $x = S\tilde{x}$.
2) If G has a negative cycle:
   a) we may assume that G is connected, since each connected component gives a separate system;
   b) construct the double cover $\bar{G}$ of G (Definition 25.7);
   c) solve the Laplacian system of the double cover, as described in Claims 25.10 and 25.11.

25.2 Double cover

Definition 25.7. Let $G = (V, E, W)$ with $V = \{v_1, \dots, v_n\}$ and nonzero real edge weights w_{ij}. The *double cover* of G is the graph $\bar{G} = (\bar{V}, \bar{E}, \bar{w})$ with $\bar{V} = \{v_1, v_1', \dots, v_n, v_n'\}$ and the following edges, for each edge v_iv_j of G:

1) if $w_{ij} > 0$, the edges v_iv_j and $v_i'v_j'$, both of weight w_{ij};
2) if $w_{ij} < 0$, the edges v_iv_j' and $v_i'v_j$, both of weight $-w_{ij}$.

The two cases look as follows:



$w_{ij} > 0$ $\qquad$ $w_{ij} < 0$

All edge weights of $\bar{G}$ are positive. A walk in G that starts at v_a lifts to a walk in $\bar{G}$ that starts at v_a (or at v_a'): it stays in the same copy along positive edges and switches between the unprimed and the primed copy along negative edges.

Claim 25.8. *If G is connected and has a negative cycle, then $\bar{G}$ is connected.*

Proof. It suffices to show that v_i is connected to v_i' for every i. Indeed, since G is connected, a path from v_1 to v_i in G lifts to a path in $\bar{G}$ from v_1 to v_i or to v_i', so then every vertex of $\bar{G}$ is joined to v_1. Let C be a negative cycle and P a path from v_i to C.

Claim. *The closed walk P, C, P^R is negative.*

Its sign is $\operatorname{sign}(P)^2 \operatorname{sign}(C) = -1$. A walk in G lifts to a walk in $\bar{G}$ that changes copies exactly at the negative edges. Thus P, C, P^R, a negative closed walk at v_i, lifts to a walk from v_i to v_i'. □

Corollary 25.9. *If G is connected and has a negative cycle, then $L(\bar{G})$ has rank $2n-1$.*

Proof. By Claim 25.8, $\bar{G}$ is a connected graph on $2n$ vertices with positive weights, so its Laplacian has rank $2n-1$ by Theorem 2.13. □

For the rest of this lecture, G is connected and has a negative cycle. Let

$$G^+ = (V, E^+, w^+), \quad \text{where } e \in E^+ \text{ if } e \in E \text{ and } w(e) > 0, \text{ and } w^+(e) = w(e),$$

and let $G^- = (V, E^-, w^-)$, where $e \in E^-$ if $e \in E$ and $w(e) < 0$, and $w^-(e) = -w(e)$. Let A^+ and A^- be the adjacency matrices of G^+ and G^-, both nonnegative, and let D be diagonal with $D_{ii} = \sum_j |w_{ij}|$. Then $L \coloneqq L_G = D - A^+ + A^-$. With the vertices of $\bar{G}$ ordered $v_1, \ldots, v_n, v_1', \ldots, v_n'$,

$$\bar{A} = \begin{pmatrix} A^+ & A^- \\ A^- & A^+ \end{pmatrix}$$

is the adjacency matrix of $\bar{G}$, each vertex v_i and v_i' has degree D_{ii} in $\bar{G}$, and

$$L(\bar{G}) = \begin{pmatrix} D & 0 \\ 0 & D \end{pmatrix} - \bar{A} =: \bar{L} = L_{\bar{G}}.$$

Since $\bar{L}\begin{pmatrix} \mathbf{1} \\ \mathbf{1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, Corollary 25.9 shows that the null space of $\bar{L}$ is spanned by $(\mathbf{1}; \mathbf{1})$.

Claim 25.10. *For $x, b \in \mathbb{R}^n$, $Lx = b$ if and only if $\bar{L}\begin{pmatrix} x \\ -x \end{pmatrix} = \begin{pmatrix} b \\ -b \end{pmatrix}$.*

Proof. By the block form of $\bar{L}$,

$$\bar{L}\begin{pmatrix} x \\ -x \end{pmatrix} = \begin{pmatrix} Dx - A^+x + A^-x \\ -A^-x - Dx + A^+x \end{pmatrix} = \begin{pmatrix} Lx \\ -Lx \end{pmatrix}.$$ □

Claim 25.11. *If $\bar{L}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ -b \end{pmatrix}$ and $\begin{pmatrix} x \\ y \end{pmatrix}^\top \mathbf{1} = 0$, then $y = -x$.*

Proof. Let $J = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}$, which swaps v_i and v_i'. The block form of $\bar{L}$ gives $J\bar{L}J = \bar{L}$, so $\bar{L}(-J(x; y)) = -J(b; -b) = (b; -b)$. Hence $(x; y) + J(x; y) = (x + y; x + y)$ lies in the null space of $\bar{L}$, so $x + y = c\mathbf{1}$ for some $c \in \mathbb{R}$. Then $0 = \mathbf{1}^\top x + \mathbf{1}^\top y = cn$, so $c = 0$ and $y = -x$. □

We now compare the eigenvalues and eigenvectors of L and $\bar{L}$. First, L is nonsingular: if $Lx = 0$, then $\bar{L}(x; -x) = 0$ by Claim 25.10, so $(x; -x)$ is a multiple of $(\mathbf{1}; \mathbf{1})$, which forces $x = 0$. By the computation in the proof of Claim 25.10,

$$Lx = \lambda x \implies \bar{L}\begin{pmatrix} x \\ -x \end{pmatrix} = \lambda \begin{pmatrix} x \\ -x \end{pmatrix}. \tag{1}$$

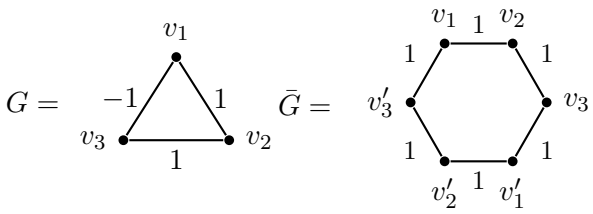
Since L is symmetric, it has n orthonormal eigenvectors, so (1) gives n orthogonal eigenvectors of $\bar{L}$.

Consider the involution f of $\bar{V}$ with $f(v_i) = v_i'$ and $f(v_i') = v_i$. It is an automorphism of $\bar{G}$: it maps the edges v_iv_j and $v_i'v_j'$ to each other, and likewise v_iv_j' and $v_i'v_j$, with the same weights. In matrix form, $J\bar{L}J = \bar{L}$. We call $(x; -x)$ an *odd* vector, since f negates it, and $(x; x)$ an *even* vector. The n eigenvectors from (1) span the n-dimensional space of odd vectors, so we have found all the odd eigenvectors. The even vectors form the orthogonal complement, which is invariant under the symmetric matrix $\bar{L}$. Thus the missing eigenvectors are even:

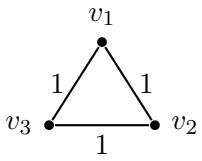
$$\bar{L}\begin{pmatrix} x \\ x \end{pmatrix} = \lambda \begin{pmatrix} x \\ x \end{pmatrix},$$

that is, $x = x'$ on the two copies. Since $\bar{L}(x; x) = \big((D - A^+ - A^-)x; (D - A^+ - A^-)x\big)$, the even eigenpairs of $\bar{L}$ are the eigenpairs of $D - A^+ - A^-$, the Laplacian of G with weights $|w_{ij}|$.

Example 25.12. Let G be the triangle with $w_{12} = w_{23} = 1$ and $w_{31} = -1$. Its double cover $\bar{G}$ is the 6-cycle $v_1v_2v_3v_1'v_2'v_3'$ with unit weights:



The even eigenpairs of $\bar{L}$ are the eigenpairs of the triangle with all weights 1:



Numerically, $\bar{L}$ has eigenvalues $0, 1, 1, 3, 3, 4$. The unit-weight triangle has eigenvalues $0, 3, 3$ (the even part), and $L = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 2 & -1 \\ 1 & -1 & 2 \end{pmatrix}$ has eigenvalues $1, 1, 4$ (the odd part).

26 Counting Spanning Trees (Matrix-Tree Theorem)

Lecture of April 10, 2020. Based on Gary L. Miller's handwritten notes.

This lecture counts spanning trees with determinants. We work with directed graphs, define divergent and convergent spanning trees and the indegree matrix, and prove the matrix-tree theorem: the number of divergent spanning trees rooted at r is the determinant of the indegree matrix with row and column r removed. The proof expands this determinant column by column into subgraphs with one edge into each vertex. For undirected graphs the theorem counts spanning trees with the Laplacian of Lecture 2, and with the matrix determinant lemma it shows that the probability that an edge lies in a random spanning tree is its effective resistance.

26.1 Random spanning tree

Let $G = (V, E)$ be a graph. We study three questions:

1) the number of spanning trees (STs) of G;

2) for an edge e, the probability $\Pr[e \in \text{random ST of } G]$;

3) a fast algorithm to find a random ST.

Definition 26.1. A *random ST* of G is a spanning tree chosen uniformly at random from all spanning trees of G.

The algorithms and results are for directed graphs, and we view undirected graphs as a special case (Lecture 26.3). Thus, until Lecture 26.3, $G = (V, E)$ is a directed graph on $V = \{v_1, \ldots, v_n\}$, also written $\{1, \ldots, n\}$, with multiple edges allowed and no self-loops.

Definition 26.2. A spanning subgraph $H \subseteq G$ is a *divergent ST* with root r if

1) r can reach all of $V(G)$ in H;

2) $\text{indeg}_H(v) = 1$ for $v \neq r$;

3) $\text{indeg}_H(r) = 0$.

It is a *convergent ST* with root r if

1) all of $V(G)$ can reach r in H;

2) $\text{outdeg}_H(v) = 1$ for $v \neq r$;

3) $\text{outdeg}_H(r) = 0$.

Definition 26.3. The *indegree matrix* of G is the $n \times n$ matrix $D = D_G$ with

$$D(i,j) = \begin{cases} \text{indeg}(i) & \text{if } i = j, \\ -k, \text{ where } k \text{ is the number of edges from } i \text{ to } j, & \text{if } i \neq j. \end{cases}$$

For $r \in V$, let $D_{r,r}$ be D with its rth row and column removed.

Since G has no self-loops, the column sums of D are zero: column j contains $\mathrm{indeg}(j)$ on the diagonal and -1 for each edge into j from another vertex. If G is undirected, viewed as a directed graph as in Lecture 26.3, then $D_G = L_G$.

Remark 26.4. The hypothesis that G has no self-loops is needed. For the graph with vertices r, v and edges $r \to v$ and $v \to v$, the loop counts in $\mathrm{indeg}(v)$, so $\det(D_{r,r}) = 2$, while G has only one divergent spanning tree rooted at r.

Lemma 26.5. *Let $r \in V$. Then G is a divergent tree with root r if and only if*

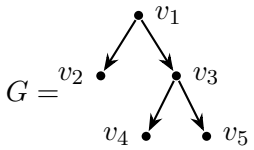
1) $D(i,i) = \begin{cases} 0 & \text{if } i = r, \\ 1 & \text{otherwise;} \end{cases}$

2) $\det(D_{r,r}) = 1$.

Moreover, if (1) holds and G is not a divergent tree with root r, then $\det(D_{r,r}) = 0$.

Proof. ($\Rightarrow$) Condition (1) holds because $D(i,i)$ is the indegree of i. For (2), order the vertices by their distance from r in G. Each vertex other than r then comes after its parent, the tail of its unique in-edge, so $D(i,j) \neq 0$ with $i \neq j$ only if i precedes j. Applying the same permutation to the rows and to the columns, which does not change determinants of principal submatrices, D becomes upper triangular with $D(i,i) = 1$ for $i \neq r$. Thus $D_{r,r}$ is upper triangular with ones on the diagonal, and $\det(D_{r,r}) = 1$.

For example, for the divergent tree with root v_1 below, $D(G)$ is already upper triangular:



$$D(G) = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & -1 & -1 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 1 & -1 & -1 \\ 4 & 0 & 0 & 0 & 1 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 \end{array}$$

($\Leftarrow$ and the last statement) Suppose that (1) holds, that is, G has indegree 1 at every vertex other than r and indegree 0 at r, but G is not a divergent tree with root r. We show that $\det(D_{r,r}) = 0$.

1) There is a vertex v that r cannot reach, since otherwise G would be a divergent tree with root r.

2) Follow the unique in-edges backward from v. This never reaches r, since r cannot reach v, so it eventually repeats a vertex. Thus G contains a cycle C with $r \notin C$, and each vertex of C has its unique in-edge from C.

3) Thus D is reducible. Order the vertices so that those of C come last. No edge enters C from outside, so $D(i,j) = 0$ for $i \notin C$ and $j \in C$, and

$$D = \begin{array}{c|cc} & V \setminus C & C \\ \hline V \setminus C & * & 0 \\ C & * & D(C) \end{array},$$

where $D(C)$ is the principal submatrix of D on C. The block with rows in C and columns outside C need not be zero, since vertices of C may have edges leaving C; so D is block lower triangular.

4) Each column of $D(C)$ contains a 1 on the diagonal and a single -1, from the in-edge inside C. Since $D(C)$ has zero column sums, $\mathbf{1}^\top D(C) = 0$ and $\det(D(C)) = 0$.

5) Since $r \notin C$, the matrix $D_{r,r}$ is still block lower triangular with the diagonal block $D(C)$, so $\det(D_{r,r}) = \det(*_{r,r}) \det(D(C)) = 0$, where $*_{r,r}$ is the upper-left block with row and column r removed. □

26.2 Main theorem

Theorem 26.6 (Matrix-tree theorem)**.** *Let G be a directed graph, with multiple edges allowed and no self-loops, and let $r \in V$. Then*

$$\#\{\textit{divergent STs of } G \textit{ rooted at } r\} = \det(D_{r,r}).$$

Proof. Without loss of generality $r = v_1$, after renaming the vertices. Recall that the column sums of D are zero.

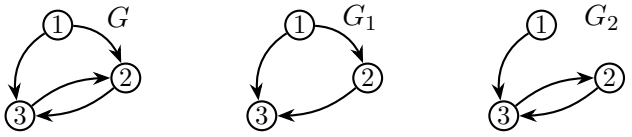
Replace the first column of D by $\chi_1 = (1, 0, \ldots, 0)^\top$, and call the result D^{new}. Expanding along the first column gives $\det(D^{\text{new}}) = \det(D_{1,1})$.

For $j \geq 2$, column j of D is the sum, over the edges e from some i into j, of the vector $\chi_j - \chi_i$. Since the determinant is linear in each column, we may expand column 2 into subgraphs with a single edge into v_2, then expand each column $j \geq 3$ in the same way. This writes $\det(D^{\text{new}})$ as a sum of $\det(M_H)$, one term for each subgraph H that has exactly one edge into each of $v_2, \ldots, v_n$ and no edge into v_1. Here M_H is the indegree matrix of H with its first column replaced by χ_1, so $\det(M_H) = \det\big((D_H)_{1,1}\big)$ by expanding along the first column. By Lemma 26.5, applied to H,

$$\det M_H = \begin{cases} 1 & \text{if } H \text{ is a divergent tree rooted at } v_1, \\ 0 & \text{otherwise.} \end{cases}$$

Every divergent ST rooted at v_1 is one of the subgraphs H, so summing gives $\det(D_{1,1}) = \det(D^{\text{new}})$ = the number of divergent STs rooted at v_1. □

Example 26.7. Let G have the vertices $1, 2, 3$ and the edges $1 \to 2$, $1 \to 3$, $3 \to 2$ and $2 \to 3$. Expanding column 2 splits G into G_1, which keeps the edge $1 \to 2$ into v_2, and G_2, which keeps $3 \to 2$:



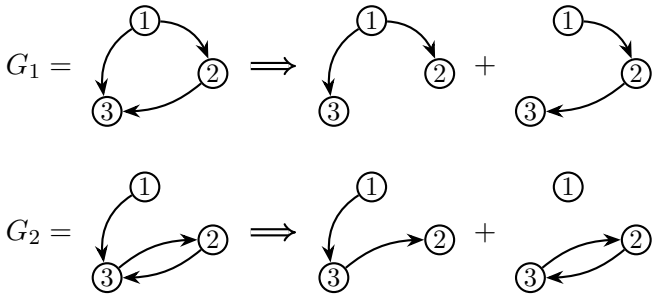
The matrices D^{new} of the three graphs are

$$G\colon \begin{pmatrix} 1 & -1 & -1 \\ 0 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}, \qquad G_1\colon \begin{pmatrix} 1 & -1 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 2 \end{pmatrix}, \qquad G_2\colon \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & -1 & 2 \end{pmatrix},$$

and by linearity in column 2,

$$\det \begin{pmatrix} 1 & -1 & -1 \\ 0 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix} = \det \begin{pmatrix} 1 & -1 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 2 \end{pmatrix} + \det \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & -1 & 2 \end{pmatrix}.$$

Expanding column 3 of each matrix in the same way gives:



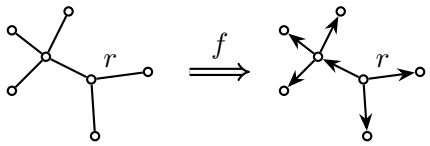
Each of the four resulting matrices M has $\det M = 1$ if its graph is a divergent tree rooted at 1, and $\det M = 0$ otherwise. Three of the four graphs are divergent trees; the last, with the cycle $2 \to 3 \to 2$, is not. Accordingly, the determinants are $3 = 2 + 1$, and G has three divergent STs rooted at 1.

26.3 The undirected case

Let $G = (V, E)$ be an undirected graph without self-loops (multiple edges allowed). We do not assume that G is connected, because the proof of Theorem 26.10 applies the results below to $G \setminus e$. Its *directed version* is $G' = (V, E')$, where each edge $\{i, j\} \in E$ gives the two edges (i, j) and (j, i) of E'. Then $D_{G'} = L_G$, the Laplacian of G.

Claim 26.8. *For all $r \in V$, $\#\mathrm{ST}(G) = \#\{$divergent STs of G' rooted at $r\}$.*

Proof. Consider the map $f\colon \mathrm{ST}(G) \to \{$divergent STs of G' rooted at $r\}$ that orients every edge of T away from r:



Claim. *f is one-to-one and onto.*

The tree $f(T)$ is a divergent ST rooted at r: every $v \neq r$ has exactly one in-edge, from its neighbor on the path from v to r in T, and r reaches every vertex along T. Forgetting the orientations recovers T, so f is one-to-one. Conversely, a divergent ST H of G' rooted at r has $n - 1$ edges, and its underlying undirected graph connects all n vertices, so it is a spanning tree T of G. Every edge of H points away from r, because the path from r to v in H is the path of T; hence $f(T) = H$, and f is onto. If G is disconnected, both sides of the claim are 0. □

Theorem 26.9. *$\#\mathrm{ST}(G) = \det(L_{1,1}) = \det(L_{r,r})$ for all $r \in V$.*

Proof. By Claim 26.8, Theorem 26.6 applied to G', and $D_{G'} = L_G$. □

For the rest of the lecture, G is connected, and the random variable T is a random ST of G. For an edge e, ER_e is the effective resistance between the endpoints of e, with unit resistances (Lectures 2 and 3).

Theorem 26.10. *For every edge e of G, $u_e \coloneqq \Pr[e \in T] = \mathrm{ER}_e$.*

Lemma 26.11 (Matrix determinant lemma)**.** *Let A be a nonsingular $n \times n$ matrix and $u, v \in \mathbb{R}^n$. Then*

$$\det(A + uv^\top) = \det(A)(1 + v^\top A^{-1} u).$$

Proof. Consider the identity

$$\begin{pmatrix} A & -u \\ v^\top & 1 \end{pmatrix} = \begin{pmatrix} A & 0 \\ v^\top & 1 \end{pmatrix} \begin{pmatrix} I & -A^{-1}u \\ 0 & 1 + v^\top A^{-1} u \end{pmatrix}. \tag{$*$}$$

We show that taking determinants in ($*$) gives the lemma. On the left, moving the last row and the last column to the front (the same permutation of rows and of columns, which keeps the determinant), and then adding u_k times the first row to row $k+1$ for each k, gives

$$\det \begin{pmatrix} A & -u \\ v^\top & 1 \end{pmatrix} = \det \begin{pmatrix} 1 & v^\top \\ -u & A \end{pmatrix} = \det \begin{pmatrix} 1 & v^\top \\ 0 & A + uv^\top \end{pmatrix} = \det(A + uv^\top).$$

On the right, the factors are block lower and block upper triangular, so the determinant is $\det(A) \cdot 1 \cdot \det(I) \cdot (1 + v^\top A^{-1} u) = \det(A)(1 + v^\top A^{-1} u)$, since $1 + v^\top A^{-1} u$ is a scalar. □

Definition 26.12. Fix a vertex r. For an $n \times n$ matrix A and a vector $x \in \mathbb{R}^n$, let $\tilde{A}$ be A with row and column r removed, and $\tilde{x}$ be x with entry r removed.

Homework 26.13. Let G be connected, $L = L_G$, and $\chi \in \mathbb{R}^n$ with $\chi \perp \mathbf{1}$. Show that $\tilde{\chi}^\top \tilde{L}^{-1} \tilde{\chi} = \chi^\top L^\dagger \chi$.

Proof of Theorem 26.10. Let $e = (i, j)$. The spanning trees of G that avoid e are the spanning trees of $G \backslash e$, and T is uniform, so $\Pr[e \notin T] = \#\mathrm{ST}(G \backslash e)/\#\mathrm{ST}(G)$. Moreover $L_{G \backslash e} = L_G - \chi_{ij}\chi_{ij}^\top$, and $\#\mathrm{ST} = \det(\tilde{L})$ by Theorem 26.9; in particular $\det(\tilde{L}_G) = \#\mathrm{ST}(G) \geq 1$, so $\tilde{L}_G$ is nonsingular. Hence

$$\begin{aligned} u_e = 1 - \Pr[e \notin T] &= 1 - \frac{\det(\widetilde{L_G - \chi_{ij}\chi_{ij}^\top})}{\det(\tilde{L}_G)} = 1 - \frac{\det(\tilde{L}_G - \tilde{\chi}_{ij}\tilde{\chi}_{ij}^\top)}{\det(\tilde{L}_G)} \\ &= 1 - \frac{\det(\tilde{L}_G)(1 - \tilde{\chi}_{ij}^\top \tilde{L}_G^{-1} \tilde{\chi}_{ij})}{\det(\tilde{L}_G)} && \text{by Lemma 26.11} \\ &= \tilde{\chi}_{ij}^\top \tilde{L}_G^{-1} \tilde{\chi}_{ij} = \chi_{ij}^\top L_G^\dagger \chi_{ij} && \text{by Homework 26.13,} \end{aligned}$$

where Lemma 26.11 is applied with $A = \tilde{L}_G$, $u = -\tilde{\chi}_{ij}$ and $v = \tilde{\chi}_{ij}$, and the two factors $\det(\tilde{L}_G)$ cancel. Finally, $\chi_{ij}^\top L_G^\dagger \chi_{ij} = \mathrm{ER}_e$ by Proposition 3.11, applied to the vector $V = L_G^\dagger \chi_{ij}$, which satisfies $L_G V = \chi_{ij}$ because $\chi_{ij} \perp \mathbf{1}$. □

27 Markov Chain Tree Theorem

Lecture of April 13, 2020. Based on Gary L. Miller's handwritten notes.

This lecture relates random walks to spanning trees. It starts from an algorithm that builds a random spanning tree of an undirected graph from a random walk, and then proves the Markov chain tree theorem: for the random walk on a strongly connected directed graph, the stationary probability of a vertex is proportional to the total weight of the convergent spanning trees rooted at it. The proof follows the tree of last exits of the walk. These trees perform a random walk of their own, and we show that the stationary probability of a tree is proportional to its weight. The directed spanning trees are those of Lecture 26.

27.1 Random spanning tree via random walks

Algorithm 27.1 (WalkTree)**. Input:** $G = (V, E)$, connected, undirected, unit weight.

1) Pick $s \in V$ arbitrarily.
2) Run a random walk from s until all vertices are visited.
3) Collect the edges $(i, j) \in E$ such that the step of the walk along (i, j) is the first visit to v_j.
4) Return the tree formed by the collected edges.

Theorem 27.2. *For every connected undirected graph G with unit weights and every choice of s, the WalkTree algorithm returns a random ST of G, that is, a spanning tree chosen uniformly at random.*

We state this theorem here without proof; Lecture 28 returns to it. For example, let G be a simple cycle C_n. Its spanning trees are the n paths obtained by deleting one edge, so the theorem says that WalkTree leaves out each edge of C_n with probability $1/n$.

27.2 Random walks 101

We use the following probability space. The *sample space* is, for example, the set of all walks on G. An *event* is, for example, the set of all walks that start at a given vertex X_0.

Definition 27.3. The probability of a walk $X_0, X_1, \ldots, X_k$ is

$$\Pr(X_0, X_1, \ldots, X_k) = \Pr(X_0) \cdot P_{X_0 X_1} \cdots P_{X_{k-1} X_k},$$

where $\Pr(X_0)$ is the probability of the start vertex and P_{ij} is the probability of a step from i to j.

A *random variable* is, for example, the *cover time*

$$C = \begin{cases} \text{the first time } t \text{ at which all of } V \text{ has been visited, starting at } X_0, \\ \text{undefined if there is no such } t. \end{cases}$$

Throughout the rest of this lecture, we assume the following.

1) $G = (V, E)$ is directed, strongly connected and unit weight, without self-loops, with $V = \{1, \ldots, n\}$ (also written $v_1, \ldots, v_n$) and $E \subseteq V \times V$.

2) All trees are convergent STs of G (Definition 26.2).

3) $P_{ij} = \text{prob}(i,j) = 1/\operatorname{outdeg}(i)$ for $(i,j) \in E$, and $P_{ij} = 0$ otherwise.

4) The weight of an edge $(i,j) \in E$ is $w_{ij} = P_{ij}$.

Definition 27.4. The *weight* of a tree T is $w(T) = \prod_{e \in T} w(e)$.

Definition 27.5. $\mathcal{T}_i(G)$ is the set of all convergent STs of G rooted at i, and $\mathcal{T}(G)$ is the set of all convergent STs of G.

We use the following standard fact about Markov chains, whose proof we omit. A *closed class* of a Markov chain on a finite set S is a set $R \subseteq S$ such that every state of R can reach every other state of R, and no state outside R can be reached from R.

Fact 27.6. *Let $(Y_t)_{t \geq 0}$ be a Markov chain on a finite set S with exactly one closed class R. Then the chain has a unique stationary distribution σ; it is positive on R and zero outside R; and for every initial distribution and every $s \in S$,*

$$\lim_{N \to \infty} \frac{1}{N} \sum_{1 \leq t \leq N} \Pr[Y_t = s] = \sigma(s).$$

27.3 Markov chain tree theorem

Theorem 27.7 (Markov chain tree theorem)**.** *The random walk on G has a unique stationary distribution π, and for every vertex i, π_i, the stationary probability of being at i, is*

$$\pi_i = \frac{\sum_{T \in \mathcal{T}_i(G)} w(T)}{\sum_{T \in \mathcal{T}(G)} w(T)}.$$

The distribution π exists and is unique by Fact 27.6: since G is strongly connected, all of V is the only closed class of the walk. (This is also a case of the Perron–Frobenius theorem, Theorem 5.13.) The rest of this lecture proves the formula for π.

Let $X_0, X_1, \ldots$ be a random walk on G, where $X_t \in \{1, \ldots, n\}$ is the index of the vertex visited at time t. Our goal is to define the *last-visited tree* B_t at time t. Let $\bar{V}_t$ be the set of indices of the vertices visited by time t, and for $i \in \bar{V}_t$ let

$$\ell(i,t) = \arg\max_{0 \leq j \leq t} \{X_j = i\}$$

be the time of the last visit to i. The tree B_t has the vertices $\bar{V}_t$ and the edges

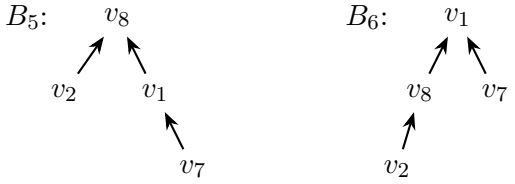
$$E(B_t) = \{(v_{X_{\ell(i,t)}}, v_{X_{\ell(i,t)+1}}) \mid i \in \bar{V}_t \setminus \{X_t\}\},$$

that is, for each visited vertex other than the current one, the edge along which the walk last left it. B_t is a convergent tree on $\bar{V}_t$ with root v_{X_t}: every vertex $i \neq X_t$ has one out-edge, to a vertex whose last visit is later than $\ell(i,t)$, so following out-edges strictly increases the time of the last visit and ends at X_t, the only vertex without an out-edge. In particular, $\text{root}(B_t) = v_{X_t}$.

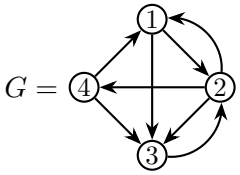
Example 27.8. Consider the walk

time t	0	1	2	3	4	5	6	7	8	9
walk X_t	2	7	1	8	2	8	1	8	2	8

At $t = 5$, the last visits are $\ell(1, 5) = 2$, $\ell(2, 5) = 4$ and $\ell(7, 5) = 1$; a vertex not visited by time 5 has no last visit and no edge in B_5. Thus B_5 has the edges $v_1 \to v_8$, $v_2 \to v_8$ and $v_7 \to v_1$, with root $v_8 = v_{X_5}$. At $t = 6$ the walk returns to v_1, and B_6 has the edges $v_8 \to v_1$, $v_7 \to v_1$ and $v_2 \to v_8$:

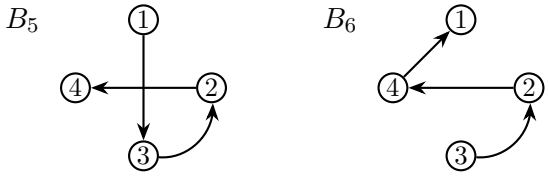


Example 27.9. Let G be the directed graph below, and consider the walk



time t	0	1	2	3	4	5	6
walk X_t	1	2	1	3	2	4	1

Every vertex has been visited by time 5, so B_5 and B_6 are convergent spanning trees, rooted at 4 and at 1:



When $\bar{V}_t = V$, the walk has reached the cover time. After the cover time, the trees $B_t, B_{t+1}, B_{t+2}, \ldots$ are rooted convergent spanning trees, and they perform a random walk on trees: if B_t is rooted at k and the walk steps from k to i, then only the last visits of k and i change, so B_{t+1} is obtained from B_t by adding (k, i) and removing the out-edge of i. Thus the next tree depends only on the current tree and the next step of the walk, and the trees B_t form a Markov chain, the *tree walk*.

Homework 27.10. Show that the tree walk has exactly one recurrent class, namely $\mathcal{T}(G)$:

1) with probability 1, all rooted trees in $\mathcal{T}(G)$ will be visited, that is, each of them equals B_t for some t;

2) from every tree in $\mathcal{T}(G)$, the tree walk reaches every tree in $\mathcal{T}(G)$ with positive probability.

Calculating the stationary distribution of the tree walk.

Definition 27.11. For $T \in \mathcal{T}(G)$, $\sigma(T)$ is the stationary probability of the tree T in the tree walk.

By Homework 27.10 and Fact 27.6, applied to the chain (B_t) whose states are the trees on subsets of V, the stationary distribution σ exists, is unique, is positive on $\mathcal{T}(G)$, and is zero on trees that do not span V.

Lemma 27.12. *For every vertex i, $\pi_i = \sum_{T \in \mathcal{T}_i} \sigma(T)$, where $\mathcal{T}_i = \mathcal{T}_i(G)$ is the set of STs rooted at i.*

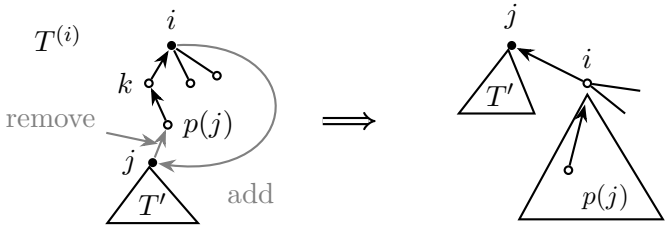
Proof. By Fact 27.6, applied first to the walk and then to the chain (B_t), and since $\mathrm{root}(B_t) = v_{X_t}$,

$$\begin{aligned}\pi_i &= \lim_{N\to\infty} \frac{1}{N} \sum_{1 \le t \le N} \Pr(X_t = i) \\ &= \lim_{N\to\infty} \frac{1}{N} \sum_{1 \le t \le N} \Pr[B_t \text{ rooted at } i] = \sum_{T \in \mathcal{T}_i} \sigma(T),\end{aligned}$$

where the last step sums the limits over the finitely many trees rooted at i, and only spanning trees have $\sigma(T) \neq 0$. □

It remains to show that $\sigma(T)$ is proportional to $w(T)$. Let $T^{(i)}$ be an ST rooted at i. We find a recurrence for $\sigma(T^{(i)})$, and for that we need the precursors of $T^{(i)}$.

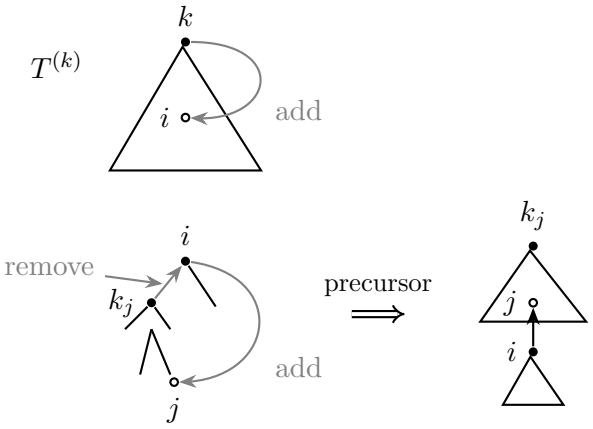
First consider a successor of $T^{(i)}$, say $T^{(j)}$, reached when the walk steps from i to j:



Thus, with $p(j)$ the head of the out-edge of j in $T^{(i)}$,

$$T^{(j)} = T^{(i)} + (i,j) - (j,p(j)).$$

Now consider a precursor of $T^{(i)}$, that is, a tree $T^{(k)}$ rooted at some k from which the step $k \to i$ of the walk leads to $T^{(i)}$:



By the successor rule, $T^{(i)} = T^{(k)} + (k,i) - (i,j)$, where $(i,j) \in E$ is the out-edge of i in $T^{(k)}$. Hence (k,i) is an edge of $T^{(i)}$, and

$$T^{(k)} = T^{(i)} + (i,j) - (k_j,i). \tag{$**$}$$

Here $k = k_j$ is determined by j. Removing (k,i) from $T^{(i)}$ and adding (i,j) gives a tree rooted at k exactly when j lies in the subtree of k, that is, when (k,i) is the last edge of the path from j to i in $T^{(i)}$. Conversely, for every $(i,j) \in E$, this vertex k_j makes ($**$) a precursor of $T^{(i)}$, and different j give different precursors, since i has different out-edges in them.

A recurrence for $T^{(i)}$. The tree walk enters $T^{(i)}$ only from its precursors, and from the precursor ($**$) with probability $P_{k_j i}$. Thus by ($**$), the stationarity of σ at $T^{(i)}$ reads

$$\sigma(T^{(i)}) = \sum_{(i,j)\in E} \sigma(T^{(i)} + (i,j) - (k_j,i)) P_{k_j i}. \tag{$*$}$$

Claim. *$w(T^{(i)})$ satisfies ($*$).*

Proof. The tree $T^{(i)} + (i,j) - (k_j,i)$ has the edge (i,j) of weight P_{ij} in place of the edge (k_j,i) of weight $P_{k_j i}$, so

$$\sum_{(i,j)\in E} w(T^{(i)} + (i,j) - (k_j,i)) P_{k_j i} = \sum_{(i,j)\in E} \frac{w(T^{(i)}) P_{ij}}{P_{k_j i}} P_{k_j i} = w(T^{(i)}),$$

where the factors $P_{k_j i}$ cancel, and the sum is $w(T^{(i)})$ since $\sum_{(i,j)\in E} P_{ij} = 1$. □

Remark 27.13. The assumption that G has no self-loops enters here. A step $i \to i$ would leave the tree unchanged, so ($*$) would need the extra term $\sigma(T^{(i)})P_{ii}$ on the right, and the sum in the claim, now over $j \neq i$, would be $1 - P_{ii}$. The weights w satisfy the equation with the extra term as well.

Proof of Theorem 27.7. The equations ($*$), one for each tree in $\mathcal{T}(G)$, are the stationarity equations of the tree walk on $\mathcal{T}(G)$, which it never leaves. By the claim, the weights $w(T) > 0$, normalized to sum to 1 over $\mathcal{T}(G)$, satisfy them, so they form a stationary distribution. Since the stationary distribution is unique, $\sigma(T) = w(T)/\sum_{T'\in\mathcal{T}(G)} w(T')$. By Lemma 27.12,

$$\pi_i = \sum_{T\in\mathcal{T}_i(G)} \sigma(T) = \frac{\sum_{T\in\mathcal{T}_i(G)} w(T)}{\sum_{T\in\mathcal{T}(G)} w(T)},$$

which is the Markov chain tree theorem. □

28 Generating Random Spanning Trees via Random Walks

Lecture of April 15, 2020. Based on Gary L. Miller's handwritten notes.

A random walk on a connected graph that runs until it has visited every vertex leaves behind a spanning tree: the edges by which it first entered the vertices. This lecture shows that, on an undirected graph, this forward tree is a uniform random spanning tree, which is the WalkTree algorithm of Lecture 27, due to Aldous [Ald90] and Broder [Bro89]. The proof compares the forward tree, built from first visits, with the last-visited tree of Lecture 27, built from last visits. Reversing time exchanges the two, and the Markov chain tree theorem (Theorem 27.7) gives the distribution of the last-visited tree.

Throughout this lecture, $G = (V, E)$ is a connected undirected graph with unit weights and $n \geq 2$ vertices, and d_i is the degree of vertex i. We identify each vertex with its index. The random walk $X_0, X_1, X_2, \ldots$ moves from i to a uniformly random neighbor, so $P_{ij} = 1/d_i$ for every edge (i, j), and X_j is the vertex visited at time j. Its stationary distribution is $\pi_1, \ldots, \pi_n$. We use the notation of Lecture 27. Spanning trees are rooted and convergent, that is, every edge points toward the root; $\mathcal{T}(G)$ is the set of all of them (Definition 27.5), $w_{ij} = P_{ij}$, and $w(T) = \prod_{e \in T} w(e)$ (Definition 27.4). The last-visited tree B_t has an edge from each vertex $i \neq X_t$ visited by time t along the edge by which the walk last left i, and it is rooted at X_t. The cover time C is the first time at which every vertex has been visited. After the cover time, $B_t, B_{t+1}, \ldots$ is a Markov chain on $\mathcal{T}(G)$ with a single recurrent class (Homework 27.10). We write $\sigma(T)$ for its stationary probability of the tree T; for every starting distribution of the walk,

$$\sigma(T) = \lim_{N \to \infty} \frac{1}{N} \sum_{0 \leq t \leq N} \Pr[B_t = T].$$

28.1 Forward trees

Definition 28.1 (Forward tree)**.** Let $t \geq 0$, and let $\bar{V}_t$ be the set of vertices visited by time t. For $i \in \bar{V}_t$, let

$$f(i, t) \coloneqq \min\{j : 0 \leq j \leq t,\ X_j = i\}$$

be the time of the first visit to i. The *forward tree* F_t has the vertex set $\bar{V}_t$ and the edges

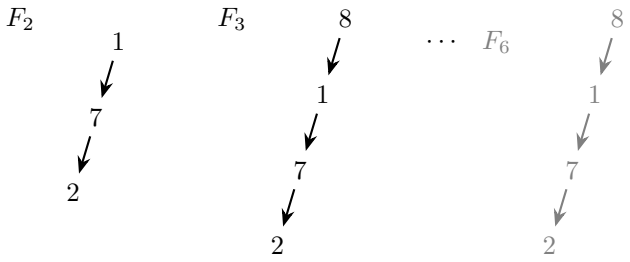
$$(X_{f(i,t)}, X_{f(i,t)-1}), \qquad i \in \bar{V}_t \setminus \{X_0\}.$$

Since $X_{f(i,t)} = i$, each visited vertex $i \neq X_0$ has exactly one edge, which leads to the vertex from which the walk first entered i. That vertex was visited earlier than i, so F_t is a tree convergent to the root X_0. The set $\bar{V}_t$ plays the role it has in the definition of B_t. After the cover time no vertex is visited for the first time, so $F_t = F_C$ for every $t \geq C$.

Example 28.2. Consider the walk

t	0	1	2	3	4	5	6	7	8	9
X_t	2	7	1	8	2	8	1	8	2	8

The tree F_2 has the edges $(7, 2)$ and $(1, 7)$, and F_3 adds the edge $(8, 1)$. The walk visits no new vertex after time 3, so $F_t = F_3$ for $3 \leq t \leq 9$; in particular $F_6 = F_3$. Edges point toward the root $X_0 = 2$.



28.2 Uniform random spanning trees from the cover time

Theorem 28.3. *Start the walk from the stationary distribution π, and let C be its cover time. Then for every $T \in \mathcal{T}(G)$,*

$$\lim_{t\to\infty} \Pr[B_t = T] = \Pr[F_C = T] = \frac{w(T)}{\sum_{T'\in\mathcal{T}(G)} w(T')}.$$

Proof. Random walks on undirected graphs are reversible:

$$P_{ij}\pi_i = P_{ji}\pi_j \qquad \text{for all } i, j.$$

Applying this once for each step, for every $k \geq 0$ and all vertices $u_0, \ldots, u_k$,

$$\Pr[X_0 = u_0, \ldots, X_k = u_k \mid \pi] = \Pr[X_0 = u_k, \ldots, X_k = u_0 \mid \pi],$$

and therefore

$$\Pr[B_k = T \mid \pi] = \Pr[F_k = T \mid \pi] \qquad \text{for every fixed } k \geq 0. \tag{28.1}$$

Indeed, reversing a walk turns each last visit into a first visit. If the last visit to $i \neq u_k$ in $u_0, \ldots, u_k$ is at time ℓ, then in the reversed walk the first visit to i is at time $k - \ell$, preceded by $u_{\ell+1}$. So B_k of the walk $u_0, \ldots, u_k$ is F_k of the reversed walk $u_k, \ldots, u_0$, with the same root u_k, and the two walks have the same probability.

Since G is connected, $C < \infty$ with probability 1, and $F_t = F_C$ for all $t \geq C$. Hence $\Pr[F_t = T \mid \pi]$ converges to $\Pr[F_C = T \mid \pi]$ as $t \to \infty$, and by (28.1) so does $\Pr[B_t = T \mid \pi]$. A Cesàro average of a convergent sequence has the same limit, so

$$\begin{aligned}
\sigma(T) &= \lim_{N\to\infty} \frac{1}{N} \sum_{0\leq t\leq N} \Pr[B_t = T \mid \pi] \\
&= \lim_{N\to\infty} \frac{1}{N} \sum_{0\leq t\leq N} \Pr[F_t = T \mid \pi] \\
&= \Pr[F_C = T \mid \pi].
\end{aligned}$$

The proof of the Markov chain tree theorem (Theorem 27.7) shows that

$$\sigma(T) = \frac{w(T)}{\sum_{T'\in\mathcal{T}(G)} w(T')}.$$

□

Remark 28.4. The limit cannot be replaced by the value at the cover time. On the path $1-2-3$, the last vertex to be visited for the first time is an end vertex, so B_C is never rooted at 2. The tree T rooted at 2, however, has $w(T) = P_{12}P_{32} = 1$, and the other two trees have weight $1/2$, so $w(T)/\sum_{T'} w(T') = 1/2$.

Corollary 28.5. *Let the random walk on G start at a vertex i, and let C be its cover time. Then F_C is a uniform random spanning tree rooted at i.*

Proof. Every forward tree is rooted at X_0. Hence for T rooted at i,

$$\Pr[F_C = T \mid X_0 = i] = \frac{\Pr[F_C = T \mid \pi]}{\pi_i},$$

which by Theorem 28.3 is proportional to $w(T)$. A tree rooted at i has exactly one edge out of each vertex $j \neq i$, and that edge has weight $P_{jk} = 1/d_j$. Hence

$$w(T) = \prod_{j \neq i} \frac{1}{d_j} = d_i \Big/ \prod_{j \in V} d_j,$$

which does not depend on T. Thus all trees rooted at i have the same probability. □

Corollary 28.6. *Under the hypotheses of Corollary 28.5, the tree F_C with the directions of its edges forgotten is a uniform random spanning tree of G.*

Proof. Directing every edge toward i is a one-to-one correspondence between the spanning trees of G and the spanning trees rooted at i, and forgetting the directions is its inverse. The claim follows from Corollary 28.5. □

29 Random Walks and Perfect Bipartite Matching

Lecture of April 17, 2020. Based on Gary L. Miller's handwritten notes.

The Ford–Fulkerson method finds a maximum flow by repeatedly finding an augmenting path in a residual graph. This lecture applies it to the simplest case, perfect matching in a regular bipartite graph, and finds each augmenting path with a random walk in the residual graph. The main result, due to Goel, Kapralov and Khanna [GKK13], is that the walks find a perfect matching in total expected time $O(n \log n)$, with no dependence on the number of edges or on the degree. We reduce this to a bound on the expected number of back edges on a single walk, and compute that number exactly by solving linear equations for these expectations.

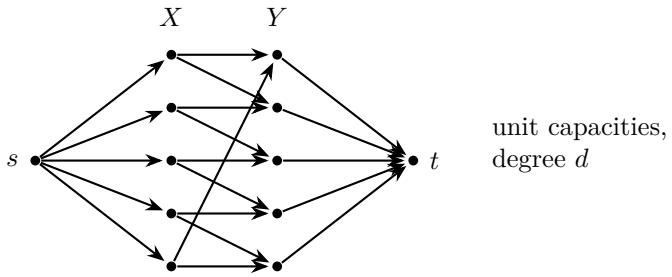
29.1 Randomized perfect bipartite matching

An *integral max-flow problem* $G = (V, E, s, t, \mathrm{cap})$ consists of a directed graph (V, E), a source s, a sink t and integral capacities $\mathrm{cap}\colon E \to \mathbb{Z}_{\geq 0}$.

Theorem 29.1 (Ford–Fulkerson)**.** *Every integral max-flow problem G has an integral maximum flow, and the value of a maximum flow equals the minimum capacity of a cut separating s from t.*

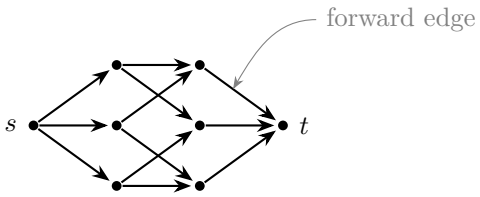
We use this classical theorem without proof. An integral max-flow problem can also have maximum flows that are not integral; the next example has one when $d \geq 2$.

Example 29.2. Let G_d be a d-regular bipartite graph with sides X and Y, where $d \geq 1$. Counting edges from each side gives $d|X| = d|Y|$, so $|X| = |Y|$. Our goal is a perfect matching, and we set this up as a flow problem: add a source s with an edge to every $x \in X$ and a sink t with an edge from every $y \in Y$, direct every edge of G_d from X to Y, and give all edges capacity 1.

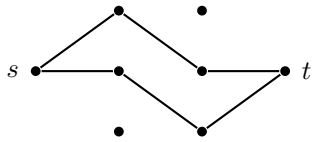


Put flow 1 on every edge out of s and on every edge into t, and flow $1/d$ on every edge between X and Y. Flow is conserved, since each vertex of X or Y has one edge of flow 1 and d edges of flow $1/d$. This flow has value $|X| = |Y|$ and saturates every edge out of s, so it is a maximum flow. By Theorem 29.1 there is also an integral flow of value $|X|$, and the edges between X and Y that carry flow 1 in it form a perfect matching. Therefore G_d has a perfect matching. The fractional flow, however, does not tell us which edges form one.

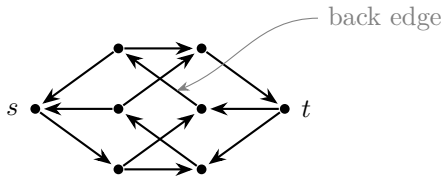
Ford–Fulkerson. Given a flow, the *residual graph* has a *forward edge* (u, v) for every edge (u, v) with unused capacity and a *back edge* (v, u) for every edge (u, v) that carries positive flow. An *augmenting path* is a path from s to t in the residual graph. With unit capacities and an integral flow, every edge is either empty and gives a forward edge, or saturated and gives a back edge. For example, take the following input with unit capacities.



Suppose that we have found two augmenting paths, each carrying one unit of flow:



This flow has the following residual graph.



Algorithm 29.3 (Ford–Fulkerson)**.** While the residual graph has an augmenting path:

1) find an augmenting path in the residual graph;

2) push flow along it and update the residual graph.

We will find each augmenting path using a random walk. A walk from s to t in the residual graph contains a path from s to t, which is an augmenting path.

29.2 The expected time of the random walks

Throughout this subsection, $G = (X \cup Y, E)$ is a connected d-regular bipartite graph, $n = |X| = |Y|$, and the flow network is that of Example 29.2. Let M be a matching of size $k < n$, viewed as an integral flow of value k, and let G_M be its residual graph. Let $X_M \subseteq X$ and $Y_M \subseteq Y$ be the matched vertices, and $X_U = X \setminus X_M$ and $Y_U = Y \setminus Y_M$ the unmatched ones, so that $|X_U| = |Y_U| = n - k$. For $y \in Y_M$, let $M(y)$ denote the match of y. The edges of G_M are: $s \to x$ for $x \in X_U$ and $x \to s$ for $x \in X_M$; $x \to y$ for each unmatched edge $(x, y) \in E \setminus M$, and $y \to x$ for each matched edge $(x, y) \in M$; $y \to t$ for $y \in Y_U$ and $t \to y$ for $y \in Y_M$ (Fig. 5).

The *random walk* in G_M starts at a vertex, leaves each vertex along a uniformly random out-edge, except that it never uses the edges $x \to s$ and $t \to y$, and stops when it reaches t. Its *back edges* are its steps $y \to M(y)$ along matched edges.

Definition 29.4. For a vertex x of G_M, let $b(x)$ be the expected number of back edges on the random walk from x to t in G_M.

Theorem 29.5. *Starting from the empty matching, run Ford–Fulkerson on the flow network of G, and find each augmenting path by a random walk from s to t in the current residual graph. Assume that the walk can leave a vertex along a uniformly random out-edge in $O(1)$ time. Then the total expected time of the walks is $O(n \log n)$.*

Remark 29.6. Neither the number of edges $|E| = dn$ nor d appears in the bound: in expectation, the walks inspect only $O(n \log n)$ edges.

Theorem 29.7. *Let M be a matching of size $k < n$. The expected number of back edges on the random walk from s to t in G_M is $b(s) \le \frac{n}{n-k} - 1$.*

We first show that Theorem 29.7 implies Theorem 29.5.

Definition 29.8. For $0 \le k < n$, let M_k be the matching of size k found by the algorithm of Theorem 29.5, and write s_k for the source s of G_{M_k}. Given M_k, let $b_k(x)$ be the expected number of back edges on the random walk from x to t in G_{M_k}, and let $\mathrm{Path}(s_k)$ be the expected number of edges on the random walk from s to t in G_{M_k}.

Proof that Theorem 29.7 implies Theorem 29.5. Let total be the total number of edges on the n walks. Each walk takes time proportional to the number of its edges, so it suffices to bound $\mathbb{E}[\text{total}]$. Given M_k, we have

$$\mathrm{Path}(s_k) = 3 + 2b_k(s) \le 3 + 2\Big(\frac{n}{n-k} - 1\Big) = 1 + \frac{2n}{n-k}.$$

Indeed, the walk uses one edge out of s, one edge into t, and one forward edge from X to Y just before t. Each back edge from Y to X is followed by one more forward edge, so each back edge costs two edges. The last equality holds because $3 + \frac{2k}{n-k} = 1 + \frac{2n-2k+2k}{n-k}$. Since this bound does not depend on M_k,

$$\mathbb{E}[\text{total}] = \sum_{k=0}^{n-1} \mathbb{E}[\mathrm{Path}(s_k)] \le \sum_{k=0}^{n-1} \Big(1 + \frac{2n}{n-k}\Big) = n + 2n \sum_{i=1}^{n} \frac{1}{i} = O(n \log n). \qquad \square$$

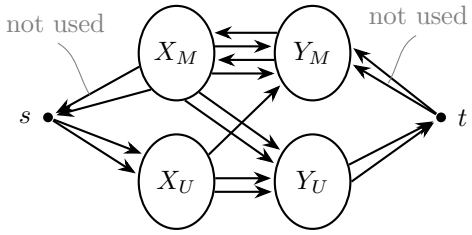
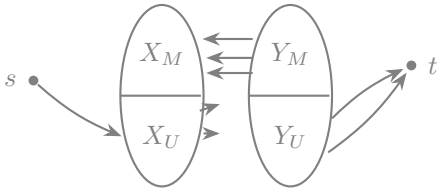


Figure 5: The residual graph G_M. Between X_M and Y_M, matched edges point from Y_M to X_M and unmatched edges from X_M to Y_M. The walk never uses the edges into s or out of t.

Proof of Theorem 29.7. The residual graph has the form shown in Fig. 5. Our goal is to bound $b(s)$; the walk from s moves through the parts of G_M as follows.



All the values $b(\cdot)$ are finite. Otherwise, since the walk is a finite Markov chain, it reaches with positive probability a nonempty set $S \subseteq X \cup Y$ that it never leaves and that misses Y_U. Every unmatched neighbor of a vertex of $S \cap X$ then lies in S, and $S \cap Y \subseteq Y_M$ has its matches in $S \cap X_M$, so $|S \cap Y| \le |S \cap X_M|$. Counting the edges between $S \cap X$ and $S \cap Y$, at most d at each vertex of $S \cap Y$, gives

$$d|S \cap Y| \ge (d-1)|S \cap X_M| + d|S \cap X_U| + |S \cap Y|.$$

Hence $S \cap X_U = \emptyset$ and equality holds throughout, so every edge of G at a vertex of S stays in S. Since G is connected, $S = X \cup Y$, which contradicts $k < n$.

We now collect equations for b. If $k \ge 1$, then $n \ge 2$, so $d \ge 2$ because G is connected.

1) If $y \in Y_U$, the walk moves to t, so
$$b(y) = 0. \tag{0}$$

2) If $y \in Y_M$, the only edge the walk can use out of y is the back edge to $M(y)$, so $b(y) = 1 + b(M(y))$. Since M is a bijection from Y_M to X_M, summing over Y and using (0) gives
$$\sum_{y \in Y} b(y) = k + \sum_{x \in X_M} b(x). \tag{1}$$

3) The walk from s moves to a uniformly random vertex of X_U, so
$$b(s) = \frac{1}{n-k} \sum_{x \in X_U} b(x). \tag{2}$$

4a) Let $x \in X_U$. All d edges at x are unmatched, so $b(x) = \frac{1}{d} \sum_{(x,y) \in E} b(y)$, that is,
$$d\, b(x) = \sum_{(x,y) \in E} b(y) \qquad (x \in X_U). \tag{3}$$

4b) Let $x \in X_M$. The walk leaves x along one of its $d-1$ unmatched edges, so $b(x) = \frac{1}{d-1} \sum_{(x,y) \in E \setminus M} b(y)$, that is,
$$(d-1)\, b(x) = \sum_{(x,y) \in E \setminus M} b(y). \tag{4}$$

If $(x,y) \in M$, then $b(y) = 1 + b(x)$ by 2), that is,
$$1 + b(x) = b(y). \tag{5}$$

Hence

$$\begin{aligned} d\, b(x) + 1 &= (d-1)\, b(x) + b(x) + 1 \\ &= \sum_{(x,y) \in E \setminus M} b(y) + b(x) + 1 && \text{by (4)} \\ &= \sum_{(x,y) \in E} b(y) && \text{by (5)}, \end{aligned}$$

that is,
$$d\, b(x) = \sum_{(x,y) \in E} b(y) - 1 \qquad (x \in X_M). \tag{6}$$

Adding (6) over X_M and (3) over X_U, and using that every $y \in Y$ has d neighbors, we get

$$\begin{aligned} d\sum_{x\in X} b(x) &= d\sum_{x\in X_M} b(x) + d\sum_{x\in X_U} b(x) \\ &= -k + \sum_{x\in X}\sum_{(x,y)\in E} b(y) \\ &= -k + d\sum_{y\in Y} b(y). \end{aligned} \tag{7}$$

By (1), this becomes

$$\begin{aligned} d\sum_{x\in X} b(x) &= -k + d\Big(k + \sum_{x\in X_M} b(x)\Big) \\ &= (d-1)k + d\sum_{x\in X_M} b(x). \end{aligned} \tag{8}$$

Subtracting $d\sum_{x\in X_M} b(x)$, which is finite, from both sides of (8) gives

$$d\sum_{x\in X_U} b(x) = (d-1)k. \tag{9}$$

Finally, by (2) and (9),

$$b(s) = \frac{1}{n-k}\sum_{x\in X_U} b(x) = \frac{(d-1)k}{d(n-k)} \le \frac{k}{n-k} = \frac{n}{n-k} - 1. \qquad \square$$

Remark 29.9. Connectivity is used only to make b finite. For a disconnected G, the walk from s never enters a component without unmatched vertices, and the same computation on the other components gives $b(s) = \frac{(d-1)k'}{d(n-k)} \le \frac{n}{n-k} - 1$, where $k' \le k$ counts the matched edges in those components. So Theorems 29.5 and 29.7 hold for every d-regular bipartite graph.

30 Guest: Machine Learning Basics for Graph Neural Networks

Lecture of April 20, 2020, the first of three guest lectures by Zico Kolter. Transcribed from his handwritten notes.

This lecture sets up supervised learning in the form that the next two lectures apply to graphs. We define the data, the hypothesis functions and the loss functions, and we state the training problem as the minimization of an average loss. We solve it by gradient descent and by minibatch stochastic gradient descent, and we compute the gradient of the logistic loss for a linear hypothesis. The same question for a two-layer neural network leads to backpropagation, the first topic of Lecture 31.

30.1 Machine learning and graph neural networks

Three guest lectures by Zico Kolter cover the following topics:

1) an introduction to machine learning, with connections to graph methods;

2) neural networks;

3) graph neural networks: graph convolutional networks and the graph transformer.

30.2 Background on machine learning

30.2.1 Supervised learning

In *supervised learning* we are given *training data*: pairs $(x^{(i)}, y^{(i)})$, $i = 1, \dots, m$, of an input $x^{(i)} \in \mathcal{X}$ and the corresponding output $y^{(i)} \in \mathcal{Y}$. The task is to learn a function $h \colon \mathcal{X} \to \mathcal{Y}$ that predicts the output of an input.

Example 30.1. In digit recognition, $x \in \mathbb{R}^{28 \times 28}$ is the image of a digit, given by its pixel values, and $y \in \{0, \dots, 9\}$ is its class label.

Most of machine learning assumes that the pairs $(x^{(i)}, y^{(i)})$ are independent and identically distributed (i.i.d.), that is, drawn independently from one distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$. Instead, we consider the case in which the pairs are the nodes of a graph and only some of the nodes carry an output (Fig. 6). This is the *semi-supervised setting*.

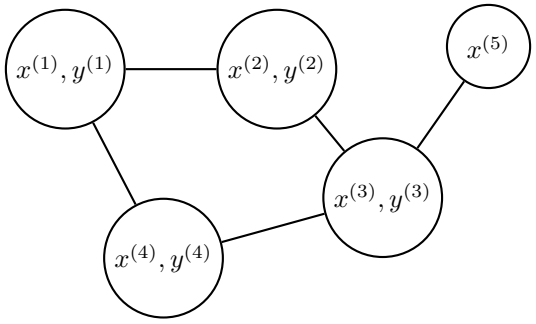


Figure 6: Data points as the nodes of a graph. Node 5 has an input $x^{(5)}$ but no label.

30.2.2 Notation and the learning problem

Throughout the three lectures, the inputs are vectors $x^{(i)} \in \mathbb{R}^n$, $i = 1, \dots, m$: here m is the number of training points and n is the dimension of the input data. (Elsewhere in the course, m and n are the numbers of edges and vertices of a graph.) The outputs are labels $y^{(i)} \in \{-1, +1\}$.

Definition 30.2. A *hypothesis function* is a map $h_\theta \colon \mathbb{R}^n \to \mathbb{R}$ that depends on a vector of *parameters* $\theta \in \mathbb{R}^k$. The value $h_\theta(x)$ is the confidence of a positive or a negative label: the prediction is good when $h_\theta(x) > 0$ for $y = +1$ and $h_\theta(x) < 0$ for $y = -1$.

Example 30.3. The *linear hypothesis* is $h_\theta(x) = \theta^\top x$ with $\theta \in \mathbb{R}^n$. A *two-layer neural network* with d hidden units is

$$h_\theta(x) = W_2\, \sigma(W_1 x + b_1) + b_2, \qquad \theta = \{W_1 \in \mathbb{R}^{d \times n},\ W_2 \in \mathbb{R}^{1 \times d},\ b_1 \in \mathbb{R}^d,\ b_2 \in \mathbb{R}\},$$

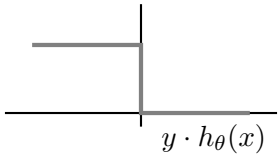
where $\sigma \colon \mathbb{R} \to \mathbb{R}$ is a nonlinearity applied entrywise, for example the ReLU, $\sigma(z) = \max\{z, 0\}$.

Definition 30.4. A *loss function* is a map $\ell \colon \mathbb{R} \times \{-1, +1\} \to \mathbb{R}_{\geq 0}$. The value $\ell(h_\theta(x), y)$ measures how well the hypothesis predicts y at x; smaller is better.

Example 30.5. The 0/1 *loss* is

$$\ell^{0/1}(h_\theta(x), y) = \begin{cases} 0 & \text{if } \operatorname{sign}(h_\theta(x)) = y, \\ 1 & \text{otherwise.} \end{cases}$$

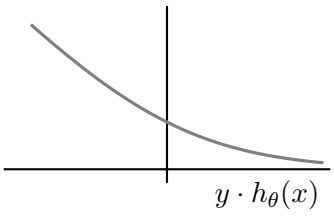
As a function of the product $y \cdot h_\theta(x)$, it is a step:



The 0/1 loss is a poor training objective for two reasons. Minimizing its average over a training set is NP-hard in general [JP78]; we state this without proof. It is also not differentiable, so gradient methods do not apply to it. Instead, one uses the *logistic loss*

$$\ell^{\text{logistic}}(h_\theta(x), y) = \log(1 + \exp(-y \cdot h_\theta(x))),$$

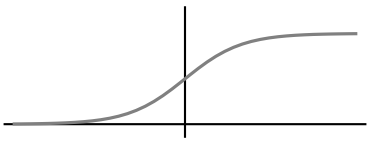
which, as a function of $y \cdot h_\theta(x)$, decreases smoothly:



By the chain rule, its derivative with respect to θ is

$$\frac{\partial \ell}{\partial \theta} = \frac{\exp(-y \cdot h_\theta(x))}{1 + \exp(-y \cdot h_\theta(x))} \cdot (-y) \cdot \frac{\partial h_\theta(x)}{\partial \theta} = \sigma(-y \cdot h_\theta(x)) \cdot (-y) \cdot \frac{\partial h_\theta(x)}{\partial \theta}, \tag{30.1}$$

where $\sigma(t) = 1/(1 + \exp(-t))$ is the *sigmoid function*. (In this formula σ is the sigmoid, not the ReLU.) The second equality holds because $e^{-z}/(1 + e^{-z}) = 1/(e^z + 1) = \sigma(-z)$. The sigmoid increases from 0 to 1:



The canonical problem of machine learning is to choose the parameters by minimizing the average loss on the training data:

$$\underset{\theta}{\text{minimize}} \ \frac{1}{m}\sum_{i=1}^{m} \ell\big(h_\theta(x^{(i)}), y^{(i)}\big). \tag{$*$}$$

What we really want is a small loss on new data from the same distribution:

$$\underset{\theta}{\text{minimize}} \ \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[\ell(h_\theta(x), y)\big].$$

Lecture 31 returns to the difference between the two problems.

30.3 Gradient descent

We solve ($*$) by gradient descent. With a step size $\alpha > 0$, it repeats the update

$$\theta := \theta - \alpha\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} \ell\big(h_\theta(x^{(i)}), y^{(i)}\big).$$

Each step uses all m training points. The following variant uses only a few of them per step.

Algorithm 30.6 (Minibatch stochastic gradient descent)**.** Input: the training data, a step size $\alpha > 0$, a batch size $p \le m$ (for example, $p = 64$), and an initial θ. Repeat:

1) Select a minibatch $B \subseteq \{(x^{(i)}, y^{(i)}) : i = 1, \dots, m\}$ with $|B| = p$, uniformly at random and without replacement.

2) Take a gradient step with respect to the gradient on B only:

$$\theta := \theta - \alpha\,\nabla_\theta \frac{1}{p}\sum_{(x,y)\in B} \ell(h_\theta(x), y).$$

The gradient in step 2 is a stochastic approximation of the full gradient. Each training pair lies in B with probability p/m, so the expectation of this gradient over the choice of B is exactly the full gradient.

30.3.1 Computing the gradient

The gradient is linear, so

$$\nabla_\theta \frac{1}{p}\sum_{(x,y)\in B} \ell(h_\theta(x), y) = \frac{1}{p}\sum_{(x,y)\in B} \nabla_\theta \ell(h_\theta(x), y),$$

and it suffices to compute the gradient of the loss at a single pair (x, y).

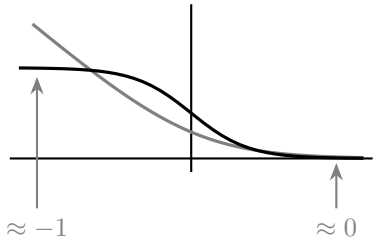
Example 30.7 (Logistic regression)**.** Let

$$\ell(h_\theta(x), y) = \log(1 + \exp(-y \cdot h_\theta(x))), \qquad h_\theta(x) = \theta^\top x, \qquad x, \theta \in \mathbb{R}^n.$$

Then

$$\begin{aligned}\nabla_\theta \log(1 + \exp(-y \cdot \theta^\top x)) &= \frac{1}{1 + \exp(-y \cdot \theta^\top x)} \cdot \exp(-y \cdot \theta^\top x) \cdot \nabla_\theta(-y \cdot \theta^\top x) \\ &= \frac{\exp(-y \cdot \theta^\top x)}{1 + \exp(-y \cdot \theta^\top x)} \cdot (-y \cdot x) \\ &= \underbrace{\frac{1}{1 + \exp(y \cdot \theta^\top x)}}_{\sigma(-y \cdot \theta^\top x)} \cdot (-y \cdot x).\end{aligned}$$

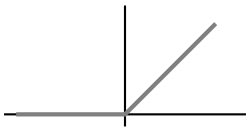
Write $z = y \cdot \theta^\top x$. The sketch below shows the logistic loss $\log(1 + e^{-z})$, the gray curve, which grows without bound on the left, and the factor $\sigma(-z)$, the black curve, which falls from about 1 on the left to 0 on the right. The two labels ≈ -1 and ≈ 0, drawn in gray with their arrows, are the slope of the loss, $\frac{d}{dz} \log(1 + e^{-z}) = -\sigma(-z)$, at the left and at the right.



Example 30.8 (Neural network)**.** Let ℓ be the logistic loss and let h_θ be the two-layer network of Example 30.3:

$$h_\theta(x) = W_2\, \sigma(W_1 x + b_1) + b_2, \qquad \theta = \{W_1 \in \mathbb{R}^{d \times n},\ W_2 \in \mathbb{R}^{1 \times d},\ b_1 \in \mathbb{R}^d,\ b_2 \in \mathbb{R}\},$$

where $\sigma \colon \mathbb{R} \to \mathbb{R}$, applied entrywise to a vector, is the ReLU $\sigma(z) = \max\{z, 0\}$:



To train this network, we need gradients such as $\nabla_{W_1} \ell(h_\theta(x), y)$, the $d \times n$ matrix of the partial derivatives of the loss with respect to the entries of W_1. The update

$$\theta := \theta - \alpha \frac{1}{p} \sum_{(x,y) \in B} \nabla_\theta \ell(h_\theta(x), y)$$

is shorthand for one update per block of parameters:

$$\begin{aligned}W_1 &:= W_1 - \alpha \frac{1}{p} \sum_{(x,y) \in B} \nabla_{W_1} \ell(h_\theta(x), y), \\ W_2 &:= W_2 - \alpha \frac{1}{p} \sum_{(x,y) \in B} \nabla_{W_2} \ell(h_\theta(x), y),\end{aligned}$$

and likewise for b_1 and b_2. These gradients are computed by *backpropagation*: the chain rule of multivariate calculus, together with some intelligent caching of intermediate results (Lecture 31).

Fact 30.9 (Universal approximation)**.** *The two-layer network of Example 30.8 is a* universal function approximator*: for every continuous function f on a compact set $K \subseteq \mathbb{R}^n$ and every $\varepsilon > 0$, there exist d and parameters θ with $|h_\theta(x) - f(x)| < \varepsilon$ for all $x \in K$. We state this without proof; see [LLPS93].*

31 Guest: Backpropagation, Generalization, Regularization, ML on Graphs

Lecture of April 22, 2020, the second of three guest lectures by Zico Kolter. Transcribed from his handwritten notes.

This second guest lecture continues Lecture 30. We first compute the gradient of the loss of a neural network by the chain rule, organized as backpropagation. We then separate the training loss from the expected loss, which is what generalization is about, and penalize $\|\theta\|_2^2$ in the training problem. Finally, we set up machine learning on graphs, where a penalty built from the normalized Laplacian encourages the predictions to vary smoothly along the edges. Graph convolutional networks, the last item of the outline for this lecture, are the subject of Lecture 32.

31.1 Backpropagation

Our goal is to compute the gradient of the loss of the two-layer network of Example 30.8 with respect to b_1:

$$\nabla_{b_1} \ell(h_\theta(x), y) = \nabla_{b_1} \log\Big(1 + \exp\big(-y \cdot (W_2\, \sigma(W_1 x + b_1) + b_2)\big)\Big).$$

This is just the multivariate chain rule, and it can be automated completely; the automated version is called *automatic differentiation*.

More generally, consider a network with $k \geq 2$ layers:

$$z_1 \coloneqq x, \qquad z_{i+1} = \sigma(W_i z_i + b_i) \quad \text{for } i = 1, \dots, k-1, \qquad h_\theta(x) = z_{k+1} = W_k z_k + b_k. \tag{31.1}$$

The hidden layers $z_2, \dots, z_k$ lie in $\mathbb{R}^d$, so $W_1 \in \mathbb{R}^{d \times n}$, $W_i \in \mathbb{R}^{d \times d}$ for $2 \leq i \leq k-1$, $W_k \in \mathbb{R}^{1 \times d}$, $b_i \in \mathbb{R}^d$ for $i < k$, and $b_k \in \mathbb{R}$; in particular, z_{k+1} is a scalar. As in the two-layer network, the last layer has no nonlinearity, and for $k = 2$ the network (31.1) is exactly that network. The nonlinearity σ is differentiable. We collect the entries of all the W_i and b_i in one parameter vector $\theta \in \mathbb{R}^N$ and want $\nabla_\theta \ell(h_\theta(x), y)$.

Definition 31.1 (Jacobian)**.** Let $f \colon \mathbb{R}^p \to \mathbb{R}^q$ be differentiable at $\theta \in \mathbb{R}^p$. The *Jacobian* of f at θ is the $q \times p$ matrix

$$\frac{\partial f}{\partial \theta} = \begin{bmatrix} \dfrac{\partial f_1}{\partial \theta_1} & \dfrac{\partial f_1}{\partial \theta_2} & \cdots \\ \dfrac{\partial f_2}{\partial \theta_1} & & \\ \vdots & & \end{bmatrix}, \qquad \text{that is,} \qquad \Big(\frac{\partial f}{\partial \theta}\Big)_{ij} = \frac{\partial f_i}{\partial \theta_j}.$$

For $q = 1$, the Jacobian is the row vector $\partial f / \partial \theta = (\nabla_\theta f)^\top$.

The chain rule says that Jacobians multiply under composition. Hence, for a parameter vector θ of one of the layers $1, \dots, k-1$, which reaches z_{k+1} only through z_k,

$$\frac{\partial \ell}{\partial \theta} = \frac{\partial \ell}{\partial z_{k+1}} \cdot \frac{\partial z_{k+1}}{\partial z_k} \cdot \frac{\partial z_k}{\partial \theta}.$$

The middle factors are computed one layer at a time. For $i < k$, write

$$z_{i+1} = \sigma(\tilde{z}_{i+1}), \qquad \tilde{z}_{i+1} \coloneqq W_i z_i + b_i;$$

since σ acts entrywise,

$$\frac{\partial z_{i+1}}{\partial z_i} = \underbrace{\frac{\partial z_{i+1}}{\partial \tilde{z}_{i+1}}}_{\operatorname{diag}(\sigma'(\tilde{z}_{i+1}))} \cdot \underbrace{\frac{\partial \tilde{z}_{i+1}}{\partial z_i}}_{W_i}.$$

For the last layer, $\partial z_{k+1}/\partial z_k = W_k$. The same reasoning, applied once per layer, gives the gradient with respect to the parameters of every layer.

Proposition 31.2. *Let $1 \le j \le k$, and let $\theta_j \in \mathbb{R}^{N_j}$ be the vector of the entries of W_j and b_j. Then*

$$\frac{\partial \ell}{\partial \theta_j} = \frac{\partial \ell}{\partial z_{k+1}} \cdot \frac{\partial z_{k+1}}{\partial z_k} \cdots \frac{\partial z_{j+2}}{\partial z_{j+1}} \cdot \frac{\partial z_{j+1}}{\partial \theta_j},$$

where for $j = k$ the right side is $\frac{\partial \ell}{\partial z_{k+1}} \cdot \frac{\partial z_{k+1}}{\partial \theta_k}$.

Proof. The parameters θ_j enter the computation (31.1) only in z_{j+1}, and for $i > j$ the vector z_{i+1} depends on θ_j only through z_i. Applying the chain rule to the composition of $\theta_j \mapsto z_{j+1}$, the maps $z_i \mapsto z_{i+1}$ for $i = j+1, \dots, k$, and $z_{k+1} \mapsto \ell$ gives the product. □

With the sizes of the factors marked, for $1 \le j < k$ (when $j = k-1$, the $d \times d$ factors are absent),

$$\underbrace{\frac{\partial \ell}{\partial \theta_j}}_{1\times N_j} = \underbrace{\frac{\partial \ell}{\partial z_{k+1}}}_{1\times 1} \cdot \underbrace{\frac{\partial z_{k+1}}{\partial z_k}}_{1\times d} \cdot \underbrace{\frac{\partial z_k}{\partial z_{k-1}}}_{d\times d} \cdots \underbrace{\frac{\partial z_{j+1}}{\partial \theta_j}}_{d\times N_j}.$$

We evaluate this product from left to right. After every factor but the first and the last, the partial product is a row vector of length d: by the chain rule, it is $\partial \ell/\partial z_i$ for the current index i. So every step multiplies a row vector by a Jacobian, a *vector-Jacobian product*, and no two Jacobians are ever multiplied together. This is *reverse-mode automatic differentiation*. It caches some intermediate results as needed: the forward pass stores the vectors $\tilde{z}_{i+1}$ that the factors $\operatorname{diag}(\sigma'(\tilde{z}_{i+1}))$ need, and the row vectors $\partial \ell/\partial z_{k+1}, \partial \ell/\partial z_k, \dots$ are computed once and serve every layer, since by Proposition 31.2 $\partial \ell/\partial \theta_j$ is $\partial \ell/\partial z_{j+1}$ times $\partial z_{j+1}/\partial \theta_j$.

31.2 Generalization

We train by solving the training problem

$$\operatorname*{minimize}_{\theta} \ \frac{1}{m} \sum_{i=1}^{m} \ell\big(h_\theta(x^{(i)}), y^{(i)}\big),$$

but its value is a bad evaluation metric. What we want is

$$\operatorname*{minimize}_{\theta} \ \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[\ell(h_\theta(x), y)\big].$$

To measure this, we draw new data $(x'^{(i)}, y'^{(i)})$, $i = 1, \dots, m'$, independently from $\mathcal{D}$: the *test set*. A better evaluation metric is the *test loss*

$$\frac{1}{m'} \sum_{i=1}^{m'} \ell\big(h_\theta(x'^{(i)}), y'^{(i)}\big).$$

The training loss is a poor estimate of the expected loss because θ was chosen to make it small. The parameters do not depend on the test set, so the expectation of the test loss over the draw of the test set equals $\mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(h_\theta(x), y)]$.

31.3 Regularization

Regularization trades off the loss on the training set against the complexity of h_θ. We measure the complexity by $\|\theta\|_2^2$ and, for a parameter $\lambda > 0$, solve

$$\underset{\theta}{\text{minimize}} \ \frac{1}{m}\sum_{i=1}^{m} \ell\big(h_\theta(x^{(i)}), y^{(i)}\big) + \lambda\|\theta\|_2^2.$$

The gradient of $\lambda\|\theta\|_2^2$ is $2\lambda\theta$, so a gradient step on one pair (x, y) becomes

$$\begin{aligned}\theta &\coloneqq \theta - \alpha\big(\nabla_\theta \ell(h_\theta(x), y) + 2\lambda\theta\big)\\ &= (1 - 2\alpha\lambda)\,\theta - \alpha\,\nabla_\theta \ell(h_\theta(x), y).\end{aligned}$$

Neural networks typically do not need regularization. They have many more parameters than data points, and they do overfit the training data; for reasons that are not understood, their test performance is still good.

31.4 ML on graphs

In machine learning on graphs, the data are inputs $x^{(i)} \in \mathbb{R}^n$ and outputs $y^{(i)} \in \{-1, +1\}$, $i = 1, \dots, m$, together with an undirected graph $G = (V, E)$ on the vertex set $V = \{1, \dots, m\}$, one vertex per data point. We represent G by its symmetric adjacency matrix $A \in \mathbb{R}_{\geq 0}^{m\times m}$.

In the *semi-supervised* variant, every vertex has an input $x^{(i)} \in \mathbb{R}^n$, but the outputs $y^{(i)}$ are known only for i in a set $\mathcal{Y} \subseteq \{1, \dots, m\}$ of labeled vertices. The hypothesis is still a map $h_\theta \colon \mathbb{R}^n \to \mathbb{R}$.

31.4.1 Laplacian regularization

To use the graph, we add to the training problem a penalty that is small when h_θ is smooth on G:

$$\underset{\theta}{\text{minimize}} \ \frac{1}{m}\sum_{i=1}^{m} \ell\big(h_\theta(x^{(i)}), y^{(i)}\big) + \lambda\, g(\text{smoothness of } h_\theta \text{ on } G).$$

The predictions $h_\theta(x^{(i)})$ sit on the vertices of G (Fig. 7).

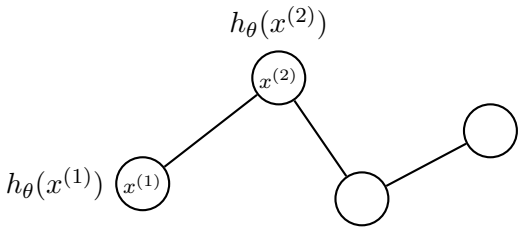


Figure 7: Predictions $h_\theta(x^{(i)})$ sit on the nodes of G.

To make the penalty concrete, stack the predictions at all the inputs, written X collectively, into a vector, and take the normalized Laplacian of G:

$$h = h_\theta(X) \coloneqq \begin{bmatrix} h_\theta(x^{(1)}) \\ h_\theta(x^{(2)}) \\ \vdots \\ h_\theta(x^{(m)}) \end{bmatrix} \in \mathbb{R}^m, \qquad L = I - D^{-1/2} A D^{-1/2}.$$

Here D is the diagonal matrix of degrees, $D_{ii} = \sum_j A_{ij}$, and L is the matrix written $\mathcal{L}$ in other lectures. We assume that every vertex has positive degree, so that $D^{-1/2}$ exists. We then solve

$$\underset{\theta}{\text{minimize}} \ \frac{1}{m} \sum_{i=1}^{m} \ell\big(h_\theta(x^{(i)}), y^{(i)}\big) + \lambda\, h^\top L h.$$

Lemma 31.3. *For every $h \in \mathbb{R}^m$,*

$$h^\top L h = \sum_{i<j} A_{ij} \Big(\frac{h_i}{\sqrt{D_{ii}}} - \frac{h_j}{\sqrt{D_{jj}}} \Big)^2 \geq 0.$$

If G is connected, then $h^\top L h = 0$ if and only if h is a multiple of $D^{1/2}\mathbf{1}$.

Proof. Set $u = D^{-1/2} h$. By the symmetry of A,

$$h^\top L h = h^\top h - u^\top A u = \sum_{i,j} A_{ij} u_i^2 - \sum_{i,j} A_{ij} u_i u_j = \frac{1}{2} \sum_{i,j} A_{ij} (u_i - u_j)^2,$$

which is the stated sum. It vanishes if and only if $u_i = u_j$ for every edge $\{i, j\}$. If G is connected, this means that u is a multiple of $\mathbf{1}$, that is, $h = D^{1/2} u$ is a multiple of $D^{1/2}\mathbf{1}$. □

The parameter λ interpolates between two extremes. As $\lambda \to 0$, the predictions $h_\theta(x^{(i)})$ tend to the traditional ML predictions, those of the problem without the penalty. As $\lambda \to \infty$, the penalty forces $h^\top L h$ toward 0, so by Lemma 31.3 (for G connected) $h_\theta(x^{(i)}) \to c\sqrt{D_{ii}}$ for all i, for some constant c. This limit is a constant vector only when G is regular. For intermediate λ, the penalty encourages h to be a smooth function on G, in the hope of better generalization.

31.4.2 Semi-supervised setting revisited

In the semi-supervised setting, only the labeled vertices contribute to the loss:

$$\underset{\theta}{\text{minimize}} \ \frac{1}{|\mathcal{Y}|} \sum_{i \in \mathcal{Y}} \ell\big(h_\theta(x^{(i)}), y^{(i)}\big) + \lambda\, h^\top L h.$$

Here $h \in \mathbb{R}^m$ is the full vector of predictions, with $h_i = h_\theta(x^{(i)})$ also for the unlabeled vertices $i \notin \mathcal{Y}$, which enter only through the penalty. It is also possible to let the entries h_i float away from $h_\theta(x^{(i)})$, treating h as a separate variable, to encourage additional smoothness.

32 Guest: Graph Neural Network Models (GCN)

Lecture of April 24, 2020, the third of three guest lectures by Zico Kolter. Transcribed from his handwritten notes.

This last guest lecture combines the neural networks of Lecture 30 with the graphs of Lecture 31. We write a neural network so that it processes all the nodes of a graph at once, and we obtain the graph convolutional network (GCN) by mixing, in every layer, the vectors at each node with those at its neighbors. We then explain the name: graph convolutions are defined through the eigenvectors of the normalized Laplacian, and their polynomial approximation, due to Defferrard, Bresson and Vandergheynst, and its first-order simplification, due to Kipf and Welling, lead back to the GCN layer. The lecture ends with two experiments, one with random weights and one on molecules.

32.1 Graph neural networks

A graph neural network has two ingredients: it is a neural network, and it uses the graph structure. The main example is the graph convolutional network.

As a first step, we write the two-layer network of Lecture 30 for all the examples at once. For one example, $h_\theta(x^{(i)})$ is a scalar and σ acts entrywise, so transposing gives

$$h_\theta(x^{(i)}) = W_2\,\sigma(W_1 x^{(i)} + b_1) + b_2 = \sigma(x^{(i)\top} W_1^\top + b_1^\top) W_2^\top + b_2^\top .$$

Stack the inputs as the rows of a matrix:

$$X = \begin{bmatrix} — & x^{(1)\top} & — \\ — & x^{(2)\top} & — \\ & \vdots & \\ — & x^{(m)\top} & — \end{bmatrix} \in \mathbb{R}^{m\times n}.$$

From here on, W_1, W_2, b_1, b_2 stand for the transposes $W_1^\top, W_2^\top, b_1^\top, b_2^\top$ of the parameters of Lecture 30, so that the b_i are row vectors. Adding a row vector b to a matrix with m rows means adding it to every row, that is, adding $\mathbf{1}b$ with $\mathbf{1} \in \mathbb{R}^m$. Then

$$h_\theta(X) \coloneqq \begin{bmatrix} h_\theta(x^{(1)}) \\ h_\theta(x^{(2)}) \\ \vdots \\ h_\theta(x^{(m)}) \end{bmatrix} = \sigma(XW_1 + b_1)W_2 + b_2.$$

Layer by layer, the network computes $Z_1 = X$ and

$$Z_{i+1} = \sigma(Z_i W_i + b_i).$$

Row j of Z_{i+1} depends only on row j of Z_i, so each example is still predicted independently of the others.

32.1.1 Graph convolutional network (GCN)

The *graph convolutional network* replaces this layer by

$$Z_{i+1} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}Z_iW_i + b_i),$$

where

$$\tilde{A} = A + I, \qquad \tilde{D}_{ii} = \sum_j \tilde{A}_{ij} \quad (\text{so } \tilde{D} = D + I), \qquad Z_i = \begin{bmatrix} — \; z_i^{(1)\top} \; — \\ \vdots \\ — \; z_i^{(m)\top} \; — \end{bmatrix}.$$

Thus $\tilde{A}$ is the adjacency matrix of G with a self-loop added at every vertex, $\tilde{D}$ is its diagonal matrix of degrees, and row j of Z_i is the vector of layer i at node j. We have $Z_1 = X$, and Z_2 is a nonlinear transform of a mixing in G of linear transforms of the rows of Z_1: row j of $\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}Z_1W_1$ is a combination of the rows of Z_1W_1 at j and at the neighbors of j.

A graph neural network is a different kind of map from a normal neural network:

$$\begin{aligned} &h_\theta\colon \mathbb{R}^n \to \mathbb{R} && \text{(normal ML)},\\ &h_\theta\colon \mathbb{R}^{m\times n} \to \mathbb{R}^m && \text{(graph neural network)}, \end{aligned}$$

where the input in $\mathbb{R}^{m\times n}$ is the matrix X, and the output gives one prediction $h_\theta(X)_i$ per node. In the semi-supervised setting of Lecture 31, we solve

$$\underset{\theta}{\text{minimize}} \; \frac{1}{|\mathcal{Y}|}\sum_{i\in\mathcal{Y}} \ell(h_\theta(X)_i, y^{(i)}),$$

where h_θ is defined as follows.

Definition 32.1. Let $\tilde{P} \coloneqq \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$. A *graph convolutional network* with $k \geq 2$ layers computes

$$Z_1 = X, \qquad Z_{i+1} = \sigma(\tilde{P}Z_iW_i + \mathbf{1}b_i) \quad \text{for } i = 1,\dots,k-1, \qquad h_\theta(X) = Z_{k+1} = Z_kW_k + \mathbf{1}b_k,$$

with $W_i \in \mathbb{R}^{d_i\times d_{i+1}}$ and $b_i \in \mathbb{R}^{1\times d_{i+1}}$, where $d_1 = n$ and $d_{k+1} = 1$; the parameters θ are all the W_i and b_i. The last layer is a linear readout, without $\tilde{P}$ and σ.

Example 32.2 (Two-layer GCN)**.** The two-layer GCN is

$$h_\theta(X) = \sigma(\tilde{P}\sigma(\tilde{P}XW_1 + b_1)W_2 + b_2)W_3 + b_3,$$

the case $k = 3$ of Definition 32.1: two graph convolutional layers followed by a linear readout. Compare a *normal* two-layer neural network:

$$h_\theta(X) = \sigma(\sigma(XW_1 + b_1)W_2 + b_2)W_3 + b_3.$$

Example 32.3 (Extreme cases)**.**

1) If $\tilde{A} = I$, that is, G has no edges (recall $\tilde{A} = A + I$), then $\tilde{D} = I$ and $\tilde{P} = I$, and the GCN reduces to a normal neural network.

2) If G is fully connected, with $\tilde{A} = \mathbf{1}\mathbf{1}^\top$, then $\tilde{D} = mI$ and $\tilde{P} = \frac{1}{m}\mathbf{1}\mathbf{1}^\top$. In the product $\tilde{P}XW_1$, every row of $\tilde{P}X$ equals $\bar{x}^\top$, where $\bar{x} = \frac{1}{m}\sum_i x^{(i)}$ is the average input. Since $\tilde{P}\mathbf{1} = \mathbf{1}$, a matrix whose rows are all equal keeps this property in every later layer. Hence we get a normal neural network applied just to the average value $\bar{x}$ of the $x^{(i)}$, and every node receives the same prediction.

32.1.2 Graph convolutions

The name comes from the convolution of signals. Let $\omega = e^{2\pi\sqrt{-1}/m}$, and index the entries of vectors in $\mathbb{R}^m$ by $0, \dots, m-1$. The (cyclic) *convolution* of $x, y \in \mathbb{R}^m$ is $(x*y)_j = \sum_{l=0}^{m-1} x_l \, y_{(j-l) \bmod m}$, and the discrete Fourier transform is $\mathrm{FFT}(x)_j = \sum_{l=0}^{m-1} \omega^{-jl} x_l$. By the convolution theorem, which we use without proof,

$$x * y = \mathrm{FFT}^{-1}\big(\mathrm{FFT}(x) \cdot \mathrm{FFT}(y)\big) = \sqrt{m}\, U\big((U^* x) \circ (U^* y)\big),$$

where the product of transforms and $\circ$ are entrywise, U is the unitary Fourier basis, $U_{jl} = \omega^{jl}/\sqrt{m}$ (complex exponentials, that is, sines and cosines), and U^* is its conjugate transpose. The second equality holds because $\mathrm{FFT}(x) = \sqrt{m}\, U^* x$ and $\mathrm{FFT}^{-1}(z) = U z/\sqrt{m}$.

On a graph, the eigenvectors of the normalized Laplacian $L = I - D^{-1/2} A D^{-1/2}$ of Lecture 31 take the place of the Fourier basis. Let $L = U \Lambda U^\top$ with U orthogonal and Λ diagonal, and define, for $x, y \in \mathbb{R}^m$,

$$x *_G y \coloneqq U\big((U^\top x) \circ (U^\top y)\big).$$

As a function of x,

$$x *_G y = U \underbrace{\mathrm{diag}(U^\top y)}_{\mathrm{diag}(\theta)} U^\top x.$$

Replacing $\mathrm{diag}(U^\top y)$ by a diagonal matrix of parameters gives the filter

$$\mathrm{conv}(x; \theta) = U \,\mathrm{diag}(\theta)\, U^\top x, \qquad \theta \in \mathbb{R}^m.$$

It has m parameters, and applying it costs $O(m^2)$ operations, two multiplications by dense $m \times m$ matrices.

Defferrard, Bresson and Vandergheynst [DBV16] restrict the filter to a polynomial in Λ.

Definition 32.4. A *polynomial filter* of degree 5 takes

$$\mathrm{diag}(\theta) = \sum_{k=0}^{5} \tilde{\theta}_k \Lambda^k, \qquad \tilde{\theta} \in \mathbb{R}^6.$$

(In their work, Chebyshev polynomials of a shifted Laplacian take the place of the powers Λ^k.)

Since $U \Lambda^k U^\top = L^k$,

$$U \,\mathrm{diag}(\theta)\, U^\top x = \sum_{k=0}^{5} \tilde{\theta}_k L^k x = \tilde{\theta}_0 x + \tilde{\theta}_1 L x + \tilde{\theta}_2 L L x + \cdots,$$

and the layer is

$$Z_{i+1} = \sigma\big(U \,\mathrm{diag}(\theta)\, U^\top Z_i W_i + b_i\big).$$

Kipf and Welling [KW17] keep only the terms of degree at most 1:

$$\begin{aligned} U \,\mathrm{diag}(\theta)\, U^\top x &= \sum_{k=0}^{1} \tilde{\theta}_k L^k x \\ &= (\tilde{\theta}_0 I + \tilde{\theta}_1 L)\, x \\ &= (\tilde{\theta}_0 I + \tilde{\theta}_1 I - \tilde{\theta}_1 D^{-1/2} A D^{-1/2})\, x \\ &= ((\tilde{\theta}_0 + \tilde{\theta}_1) I + \theta_2 D^{-1/2} A D^{-1/2})\, x \\ &= \theta'(I + D^{-1/2} A D^{-1/2})\, x, \end{aligned}$$

where $\theta_2 := -\tilde{\theta}_1$, and the last step takes one shared parameter $\theta' := \tilde{\theta}_0 + \tilde{\theta}_1 = \theta_2$, that is, $\tilde{\theta}_0 = -2\tilde{\theta}_1$. Absorbing θ' into the weight matrix W_i, the layer applies $I + D^{-1/2}AD^{-1/2}$ to Z_iW_i. Finally, the identity is moved inside the normalization: Kipf and Welling replace $I+D^{-1/2}AD^{-1/2}$, whose eigenvalues lie in $[0, 2]$, by $\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ with $\tilde{A} = A + I$ (their renormalization trick), whose eigenvalues lie in $(-1, 1]$. The layer becomes

$$Z_{i+1} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}Z_iW_i + b_i),$$

the GCN layer of Definition 32.1.

32.2 Experiments

Example 32.5 (Random weights)**.** Create a three-layer GCN with *random weights*, hidden width d and $W_3 \in \mathbb{R}^{d\times 2}$, so that it maps every node to a vector in $\mathbb{R}^2$, and run it on random inputs for some graph. The embeddings produced by the network are observed to cluster close neighborhoods in the graph.

Figure 8 shows a molecule as a graph: the vertices are the atoms and the edges are the bonds,

$$V = \{\text{atoms}\}, \qquad E = \{\text{bonds}\},$$

with an edge weight w given by a distance (or by a charge, or by 1/charge). The input x at a node is the atom type and the charge. This graph was not used in the experiment.

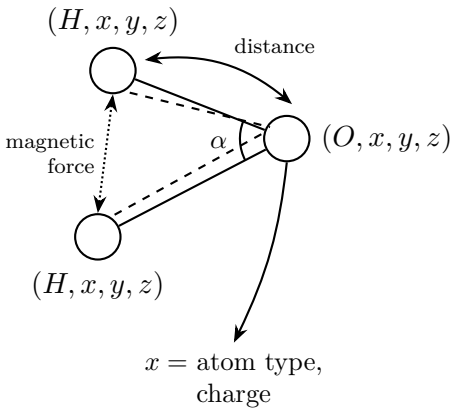


Figure 8: A molecule drawn as a graph. Each atom is labeled by its type and its coordinates.

Instead, the experiment used a *meta-graph* G whose vertices are the atoms, the bonds and the triplets (Fig. 9),

$$V = \{\text{atoms},\ \text{bonds},\ \text{triplets}\},$$

with a distance D_{ij} between its vertices that satisfies $D_{ij} = 0$ when i is an atom, j is a bond, and the atom is part of the bond. This construction embeds all the information about atoms, bonds and triplets directly into the input X. The networks used had about 16 layers with 600 hidden units, on molecules with at most 29 atoms.

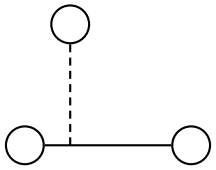


Figure 9: A sketch of the meta-graph construction.

33 Maximum Flow via Electrical Flow

Lecture of April 29, 2020. Based on Gary L. Miller's handwritten notes.

This lecture begins an approach to maximum flow through electrical flows. We view a flow as a vector in $\mathbb{R}^m$. The flows of a given value form an affine space, and the flows within capacity form a cube, so a maximum flow is a point in their intersection; the electrical flow is the point of the affine space nearest the origin, and it costs one Laplacian solve. Starting from the electrical flow, we alternate projections onto the cube and onto the affine space. We compute the projection onto the affine space with linear algebra, and show that one round of the two projections is a gradient step for the squared overflow. Lecture 34 accelerates this gradient descent with Nesterov's method.

33.1 Max flow via electric flow

Throughout this lecture, $G = (V, E)$ is an undirected graph with n vertices, m edges, unit capacities and two distinct vertices s and t that lie in the same connected component. Each edge has a fixed orientation, which only gives a sign to the flow on it: flow $f(e) < 0$ on an edge e oriented from u to v means flow $|f(e)|$ from v to u.

Definition 33.1. A *combinatorial flow* is a vector $f \in \mathbb{R}^m$, indexed by the edges, such that

1) the flow into v equals the flow out of v for every $v \in V \setminus \{s, t\}$;

2) $|f(e)| \leq 1$ for every edge e.

Its *value* F is the net flow out of s.

Our goal is a maximum flow from s to t, that is, a combinatorial flow of maximum value F.

33.1.1 Geometric view

1) The *boundary map* B is the $n \times m$ matrix whose column for the edge e, oriented from u to v, has the entry 1 in the row of the tail u and -1 in the row of the head v, and 0 elsewhere. With this sign, $(Bf)_v$ is the net flow out of v. This B is the negative of the boundary operator of Definition 2.8; the Laplacian $L = BB^\top$ is the same for both.

2) B_∞^m is the unit ℓ_∞ ball,

$$B_\infty^m \coloneqq \{x \in \mathbb{R}^m : |x_i| \leq 1 \text{ for all } i\}.$$

3) The flow value $F \geq 0$ is fixed. By binary search over F, we may assume that we know the value of a maximum flow.

4) $\chi_{st} \in \mathbb{R}^n$ has the entry 1 at s, the entry -1 at t, and 0 elsewhere.

33.1.2 Geometric goal

The maximum flow problem, for the value F, is to find $f \in \mathbb{R}^m$ with

1) $Bf = F\chi_{st}$, and

2) $f \in B_\infty^m$.

The first condition is conservation at every $v \neq s, t$ together with net flow F out of s, and the second is the capacity constraint; so the solutions are exactly the combinatorial flows of value F.

33.1.3 The electrical flow problem

Let

$$\mathcal{Z}_F \coloneqq \{f \in \mathbb{R}^m : Bf = F\chi_{st}\}.$$

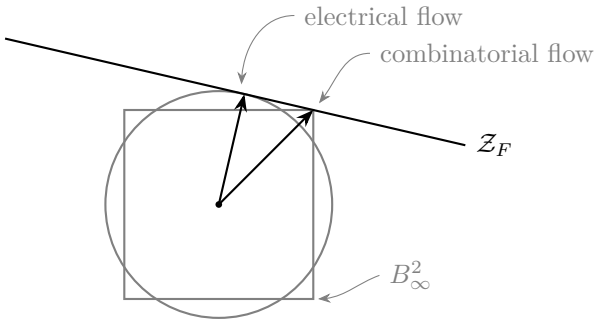
The *electrical flow problem* is to find $f \in \mathcal{Z}_F$ that minimizes $f^\top f = \|f\|_2^2$. We can solve it with one Laplacian solve:

1) solve $BB^\top v = F\chi_{st}$;

2) return $f_p = B^\top v$.

This is Thomson's principle (Theorem 3.9) with unit resistances, scaled by F: the minimizer is the electrical flow, the potential flow $B^\top v$ of value F. In more detail, $BB^\top$ is block diagonal, with one block for each connected component of G, and by Theorem 2.13 the kernel of each block is spanned by the all-ones vector of that component. Since χ_{st} sums to 0 on every component, the system is solvable. Thomson's principle applies to the component that contains s and t. On every other component, a flow in $\mathcal{Z}_F$ is a circulation, so the least norm there is attained by 0; and v is constant there, so $B^\top v = 0$ there. Likewise, a combinatorial flow remains one, of the same value, when it is set to 0 off the component of s and t. So we may delete the other components at the start and assume that G is connected.

The two problems differ only in the norm. The electrical flow is the point of $\mathcal{Z}_F$ of least ℓ_2 norm, while a combinatorial flow of value F is a point of $\mathcal{Z}_F$ of ℓ_∞ norm at most 1. For $m = 2$ the picture is the following.



Let

$$C \coloneqq \{f \in \mathbb{R}^m : Bf = 0\}$$

be the space of *circulations*. If $f_p = B^\top v$ is the electrical flow, then

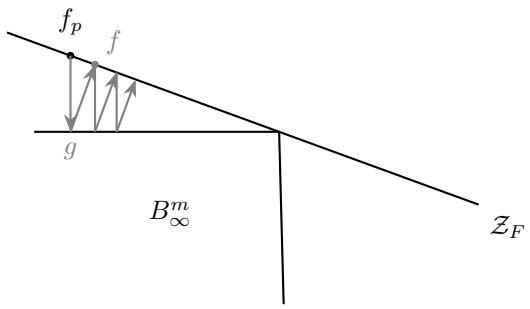
$$\mathcal{Z}_F = \{f + f_p : f \in C\}.$$

This suggests repeated projections, starting from the electrical flow:

1) let g be the projection of f_p onto B_∞^m;

2) let f be the projection of g onto $\mathcal{Z}_F$;

3) repeat, with f in place of f_p.



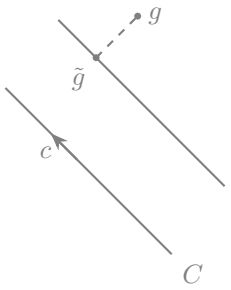
The projection onto B_∞^m is truncation, $\mathrm{Proj}(f, B_\infty^m)_e = \max\big(-1, \min(1, f(e))\big)$, so it takes linear time.

33.1.4 Linear algebra 101: projection onto an affine subspace

Our next goal is to compute the projection $\tilde{g}$ of $g \in \mathbb{R}^m$ onto $\mathcal{Z}_F$. It is the point $\tilde{g} = \mathrm{Proj}(g, \mathcal{Z}_F)$ characterized by

1) $B\tilde{g} = F\chi_{st}$, and

2) $\langle c, g - \tilde{g}\rangle = 0$ for all $c \in C$.



Every $g \in \mathbb{R}^m$ splits as follows: there exist $g_p, g_c \in \mathbb{R}^m$ with

0) $g = g_p + g_c$,

1) $g_p \perp g_c$,

2) $g_p = B^\top x$ for some $x \in \mathbb{R}^n$ (a *potential flow*),

3) $Bg_c = 0$ (a circulation).

This is the orthogonal decomposition $\mathbb{R}^m = \mathrm{col}(B^\top) \oplus \mathrm{Ker}(B)$.

Claim 33.2. *$f_p + g_c$ is the projection of g onto $\mathcal{Z}_F$.*

Proof. We check the two conditions. First,

$$B(f_p + g_c) = Bf_p + Bg_c = F\chi_{st},$$

since $Bg_c = 0$, so $f_p + g_c \in \mathcal{Z}_F$. Second,

$$g - (f_p + g_c) = g_p + g_c - f_p - g_c = g_p - f_p$$

is a potential flow, so $\langle g_p - f_p, c\rangle = 0$ for every circulation c. □

To compute g_p, let $b = Bg$. Since $Bg_c = 0$, we have $BB^\top x = b$, and hence

$$g_p = B^\top (BB^\top)^\dagger b, \qquad \text{that is,} \qquad g_p = (B^\top (BB^\top)^\dagger B)g.$$

The matrix $\Pi \coloneqq B^\top (BB^\top)^\dagger B$ is the orthogonal projection onto $\mathrm{col}(B^\top)$. Now $g_c = g - g_p = (I - \Pi)g$, and therefore

$$\mathrm{Proj}(g, \mathcal{Z}_F) = f_p + (I - \Pi)g.$$

33.1.5 Combining the two projections

Let $f \in \mathcal{Z}_F$, let $g = \mathrm{Proj}(f, B^m_\infty)$, and let $\bar{f} = \mathrm{Proj}(g, \mathcal{Z}_F)$. We decompose

$$\begin{aligned}
&1) \quad f = g + f_{\text{over}}, && g \in B^m_\infty, \\
&2) \quad f = f_p + f_c, && f_c \in C, \qquad (1)\\
&3) \quad g = g_p + g_c, \\
&4) \quad f_{\text{over}} = h_p + h_c, \\
&5) \quad f_c = g_c + h_c, \qquad (2)
\end{aligned}$$

where 1) defines the overflow f_{over}, 2) holds because $f - f_p \in C$, and 3) and 4) are the splittings above. Applying $I - \Pi$ to 1) gives 5), since $(I - \Pi)f = f_c$. By Claim 33.2,

$$\begin{aligned}
6) \quad \bar{f} &= f_p + g_c \\
&= (f - f_c) + (f_c - h_c) && \text{by (1) and (2)} \\
&= f - h_c.
\end{aligned}$$

Thus

$$\bar{f} = f - (I - \Pi) f_{\text{over}}. \qquad (*)$$

The simple algorithm repeats (∗).

Consider the objective function

$$\Phi(y) \coloneqq \tfrac{1}{2}\|y - \mathrm{Proj}(y, B^m_\infty)\|_2^2 = \tfrac{1}{2}\|\mathrm{over}(y)\|_2^2,$$

where $\mathrm{over}(y) \coloneqq y - \mathrm{Proj}(y, B^m_\infty)$ is the *overflow* of y; in particular $f_{\text{over}} = \mathrm{over}(f)$.

Claim 33.3. *For every $y \in \mathbb{R}^m$, $\nabla\Phi(y) = y - \mathrm{Proj}(y, B^m_\infty) = \mathrm{over}(y)$.*

Proof. We have $\Phi(y) = \frac{1}{2}\sum_e \varphi(y(e))$ with $\varphi(u) = \big(\max(|u| - 1, 0)\big)^2$. The function φ is continuously differentiable, with $\varphi'(u) = 2(u - 1)$ for $u > 1$, $\varphi'(u) = 2(u + 1)$ for $u < -1$, and $\varphi'(u) = 0$ otherwise. In each case $\frac{1}{2}\varphi'(u) = u - \max\big(-1, \min(1, u)\big)$, which is the coordinate e of $\mathrm{over}(y)$ at $u = y(e)$. □

For $f \in \mathcal{Z}_F$, let $\nabla_{\mathcal{Z}_F}\Phi(f)$ be the gradient of Φ restricted to $\mathcal{Z}_F$, that is, the projection of $\nabla\Phi(f)$ onto C, the space of directions parallel to $\mathcal{Z}_F$. By Claim 33.3,

$$\begin{aligned}\nabla_{\mathcal{Z}_F}\Phi(f) &= \mathrm{Proj}(\nabla\Phi(f), C)\\ &= \mathrm{Proj}(f_{\mathrm{over}}, C)\\ &= (f_{\mathrm{over}})_C\\ &= (I - \Pi)f_{\mathrm{over}},\end{aligned}$$

where $(f_{\mathrm{over}})_C = h_c$ is the circulation part of f_{over}. Hence $(*)$ reads $\bar{f} = f - \nabla_{\mathcal{Z}_F}\Phi(f)$, and our algorithm is just gradient descent with step size $1 = 1/C_L$, since $\nabla\Phi$ is 1-Lipschitz ($C_L = 1$ in the notation of Lecture 34; see Lemma 34.10).

34 Nesterov's Gradient Method and the LRS Approximate Max-Flow Algorithm

Lecture of May 1, 2020. Based on Gary L. Miller's handwritten notes.

This lecture presents Nesterov's accelerated gradient method [Nes83] and uses it to speed up the gradient descent of Lecture 33, which gives the approximate maximum flow algorithm of Lee, Rao and Srivastava [LRS13]. We define the two constants of a convex function that govern gradient methods, the Lipschitz constant of the gradient and the strong convexity constant, and compute them for quadratic functions. We state Nesterov's iteration bounds with and without strong convexity, and compare the method with Chebyshev's method (Lecture 14.6). Applied to the squared overflow with exact Laplacian solves, the bound without strong convexity yields a flow within capacity of value at least $(1-2\varepsilon)F$ after $O(\varepsilon^{-1}\sqrt{m/F})$ gradient steps, each of which costs one Laplacian solve.

34.1 Nesterov's gradient method

In this subsection and the next, $f\colon \mathbb{R}^m \to \mathbb{R}$ is a convex, differentiable function with a gradient oracle, which returns $\nabla f(y)$ for any $y \in \mathbb{R}^m$. Given the oracle, two constants of f and a starting point $y_0 \in \mathbb{R}^m$, the method computes $\min_{y\in\mathbb{R}^m} f(y)$ approximately.

Definition 34.1. The gradient of f is *C_L-Lipschitz* if

$$\|\nabla f(y) - \nabla f(z)\|_2 \le C_L \|y - z\|_2 \qquad \text{for all } y, z \in \mathbb{R}^m,$$

and the least such C_L is the *Lipschitz ∇ constant* of f. The function f is *C_{sc}-strongly convex* if

$$\langle \nabla f(y) - \nabla f(z), y - z\rangle \ge C_{sc}\|y - z\|_2^2 \qquad \text{for all } y, z \in \mathbb{R}^m,$$

and the largest such C_{sc} is the *strong convexity* constant of f.

Example 34.2. Let $f(x) = \frac{1}{2}x^\top Ax$, where A is symmetric positive definite, so that $\nabla f(x) = Ax$. Setting $x = y - z$, the Lipschitz condition becomes

$$\|Ax\|_2 \le C_L\|x\|_2 \qquad \text{for all } x \in \mathbb{R}^m,$$

so the Lipschitz ∇ constant is $C_L = \lambda_{\max}(A)$. Similarly, the strong convexity condition becomes

$$\frac{x^\top Ax}{x^\top x} \ge C_{sc} \qquad \text{for all } x \ne 0,$$

so the strong convexity constant is $C_{sc} = \lambda_{\min}(A)$.

An orthogonal change of variables $x = Qu$, with $A = Q\Lambda Q^\top$, turns f into $\frac{1}{2}u^\top \Lambda u$ and preserves norms, so it changes neither constant. Hence we may assume that A is diagonal, for example

$$A = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}, \quad a \ge b > 0, \qquad f(x) = \tfrac{1}{2}(ax_1^2 + bx_2^2), \qquad \nabla f(x) = (ax_1, bx_2),$$

with $C_L = a$ and $C_{sc} = b$.

34.2 Nesterov's algorithm

Theorem 34.3. *There exists a 2-term recurrence, Algorithm 34.6 below, with the following properties. Let the gradient of f be C_L-Lipschitz, let $y_0 \in \mathbb{R}^m$, let y^* be a minimizer of f, and let $\varepsilon > 0$.*

A) If f is C_{sc}-strongly convex with $C_{sc} > 0$, then for every integer

$$T \geq \sqrt{\frac{C_L}{C_{sc}}} \ln\Big(\frac{C_L \|y_0 - y^*\|_2^2}{\varepsilon}\Big),$$

the algorithm returns after T iterations a point y_T with $f(y_T) - f(y^*) \leq \varepsilon$.

B) Without strong convexity, run with $C_{sc} = 0$. Then for every integer

$$T \geq \sqrt{\frac{2C_L}{\varepsilon}} \, \|y_0 - y^*\|_2,$$

the algorithm returns after T iterations a point y_T with $f(y_T) - f(y^*) \leq \varepsilon$.

We state this theorem without proof. The bounds in A) and B) follow from [Bub15, Theorem 3.18] and [BT09, Theorem 4.4], respectively.

Theorem 34.4. *Let $f(x) = \frac{1}{2} x^\top A x$ with A symmetric positive definite. Nesterov's method with $C_L = \lambda_{\max}(A)$ and $C_{sc} = \lambda_{\min}(A)$ is, like Chebyshev's method, a 2-term recurrence, and both reach $f(y_k) \leq \varepsilon \lambda_{\max}(A) \|y_0\|_2^2$ within $O\big(\sqrt{\lambda_{\max}(A)/\lambda_{\min}(A)} \, \ln(1/\varepsilon)\big)$ iterations.*

Remark 34.5. The two methods are not identical. With $\kappa = \lambda_{\max}(A)/\lambda_{\min}(A)$, Chebyshev's method contracts the error along the slowest eigenvector by about $(\sqrt{\kappa}-1)/(\sqrt{\kappa}+1) \approx 1 - 2/\sqrt{\kappa}$ per step, and Nesterov's method by about $1 - 1/\sqrt{\kappa}$.

Algorithm 34.6 (Nesterov$(f, \nabla f, C_L, C_{sc}, y_0)$)**.** Set $z_1 = y_0$ and $a_1 = 1$. For $k = 1$ to T, the number of iterations:

1) Set $y_k = z_k - \dfrac{1}{C_L} \nabla f(z_k)$.

2) If $C_{sc} > 0$, set

$$z_{k+1} = y_k + \frac{\sqrt{C_L} - \sqrt{C_{sc}}}{\sqrt{C_L} + \sqrt{C_{sc}}} \, (y_k - y_{k-1}).$$

3) If $C_{sc} = 0$, set

$$a_{k+1} = \Big(1 + \sqrt{4a_k^2 + 1}\Big)\Big/2 \qquad \text{and} \qquad z_{k+1} = y_k + \frac{a_k - 1}{a_{k+1}} \, (y_k - y_{k-1}).$$

Return y_T.

34.3 The LRS approximate max flow algorithm [LRS13]

We use the notation of Lecture 33: the graph G with m edges, unit capacities and the vertices s and t; the boundary map B and the Laplacian $L = BB^\top$; the affine space $\mathcal{Z}_F$ of flows of value F and the circulations C; the box B_∞^m; the projection $\Pi = B^\top L^\dagger B$ onto potential flows, with $\Pi^\perp = I - \Pi$; the overflow $\mathrm{over}(y) = y - \mathrm{Proj}(y, B_\infty^m)$; and $\Phi(y) = \frac{1}{2}\|\mathrm{over}(y)\|_2^2$.

Algorithm 34.7 (LRS MaxFlow(G: graph; $F > 0$: flow value; $\varepsilon > 0$))**.**

1) Find the potential flow $y_0 \in \mathcal{Z}_F$ of value F (the electrical flow f_p of Lecture 33).

2) Run Nesterov with the following settings:

   a) starting from y_0;

   b) $f = \Phi(y) = \frac{1}{2}\|\mathrm{over}(y)\|_2^2$ restricted to $\mathcal{Z}_F$;

   c) compute ∇f by
   $$\nabla f(y) = \Pi^\perp \nabla\Phi(y) = (I - B^\top L^\dagger B)\,\mathrm{over}(y);$$

   d) $C_L = 1$ and $C_{sc} = 0$;

   e) $T = \left\lceil \frac{2}{\varepsilon}\sqrt{\frac{m}{F}} \right\rceil$;

   f) return $y_T \in \mathcal{Z}_F$.

3) Rescale y_T to $\tilde{y}_T = (\frac{1}{1+\varepsilon})y_T$.

4) Make $\tilde{y}_T$ into $\bar{y}_T \in B_\infty^m$ by draining off the overflow (Lecture 34.5).

5) Return $\bar{y}_T$.

Every gradient $\nabla f(y)$ lies in C, so all the iterates of Nesterov's method stay in $\mathcal{Z}_F$, and the method runs on f as a function on the affine space $\mathcal{Z}_F$.

34.4 Main theorem

Theorem 34.8. *If F is feasible, that is, if G has a combinatorial flow of value F, then LRS MaxFlow returns a combinatorial flow $\bar{y}_T$ with*

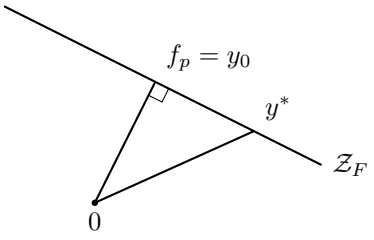
1) $\mathrm{flow}(\bar{y}_T) \geq (1 - 4\varepsilon)F$*;*

2) running time $O(\frac{1}{\varepsilon}\sqrt{\frac{m}{F}}\, m \log^2 n)$*, if G is connected.*

Proof. We prove part 1 in the stronger form $\mathrm{flow}(\bar{y}_T) \geq (1 - 2\varepsilon)F$, assuming that y_0 and $\nabla\Phi$ are computed exactly. If the Laplacian systems are solved only approximately, the algorithm still returns a flow of value at least $(1 - 4\varepsilon)F$; we do not prove this, nor the running time in part 2. Let y^* be a combinatorial flow of value F, so that $y^* \in \mathcal{Z}_F \cap B_\infty^m$.

Claim 34.9. $\|y_0 - y^*\|_2 \leq \|y^*\|_2 \leq \sqrt{m}$.

Proof. The electrical flow $y_0 = B^\top v$ is a potential flow, and $y^* - y_0 \in C$, so $\langle y_0, y^* - y_0\rangle = v^\top B(y^* - y_0) = 0$. Hence
$$\|y_0\|_2^2 + \|y_0 - y^*\|_2^2 = \|y^*\|_2^2,$$
as in the picture below, and $\|y^*\|_2 \leq \sqrt{m}$ because $|y^*(e)| \leq 1$ for every edge e.



□

Lemma 34.10. *Let $K \subseteq \mathbb{R}^m$ be a nonempty closed convex set, and let $\mathrm{Proj}(x, K)$ be the point of K nearest to x.*

1) The map $x \mapsto \mathrm{Proj}(x, K)$ is 1-Lipschitz.

2) If $\Phi(x) = \frac{1}{2}\|x - \mathrm{Proj}(x, K)\|_2^2$, then $\nabla\Phi(x) = x - \mathrm{Proj}(x, K)$, and $\nabla\Phi$ is 1-Lipschitz.

We state this lemma without proof.

Claim 34.11. *∇f is 1-Lipschitz.*

Proof. For $y \in \mathcal{Z}_F$, one gradient step of f is

$$y \mapsto y - \nabla f(y) = \mathrm{Proj}(\mathrm{Proj}(y, B_\infty^m), \mathcal{Z}_F)$$

by $(*)$, where $\nabla f(y) = (I - \Pi)(y - \mathrm{Proj}(y, B_\infty^m))$. By Lemma 34.10, part 2, with $K = B_\infty^m$, the map $y \mapsto y - \mathrm{Proj}(y, B_\infty^m)$ is 1-Lipschitz, and $I - \Pi$ is an orthogonal projection. Hence ∇f is 1-Lipschitz. □

The function f is convex, because $\Phi(y) = \frac{1}{2}\sum_e \big(\max(|y(e)| - 1, 0)\big)^2$ is a sum of convex functions of the coordinates. We now apply Nesterov's method. By part B of Theorem 34.3, with y^* a minimizer of f since $\Phi(y^*) = 0$, running for $T' \geq 2\sqrt{C_L/\varepsilon'}\,\|y_0 - y^*\|_2 = \sqrt{2C_L/(\varepsilon'/2)}\,\|y_0 - y^*\|_2$ iterations gives error at most $\varepsilon'/2$. For us $C_L = 1$ and $\|y_0 - y^*\|_2 \leq \sqrt{m}$ by Claim 34.9, so

$$T' \geq 2\sqrt{\frac{1}{\varepsilon'}}\,\sqrt{m} \quad \text{iterations give error at most } \varepsilon'/2.$$

Set $\varepsilon' = \varepsilon^2 F$. Then

$$2\sqrt{\frac{1}{\varepsilon^2 F}}\,\sqrt{m} = \frac{2}{\varepsilon}\sqrt{\frac{m}{F}} \leq T,$$

the number of iterations in Algorithm 34.7. Thus

$$\Phi(y_T) - \Phi(y^*) \leq \frac{\varepsilon^2 F}{2}, \qquad \Phi(y^*) = 0, \qquad \|\mathrm{over}(y_T)\|_2^2 = 2\Phi(y_T) \leq \varepsilon^2 F.$$

This bounds the overflow in the squared ℓ_2 norm, but draining costs its ℓ_1 norm. We now rescale to remove every overflow below ε.

Definition 34.12. $D \coloneqq \{e : |\mathrm{over}(y_T(e))| \geq \varepsilon\}$.

Claim 34.13. $|D| \leq F$.

Proof. We have

$$\varepsilon^2 F \geq \|\mathrm{over}(y_T)\|_2^2 \geq \sum_{e \in D} \mathrm{over}\big(y_T(e)\big)^2 \geq \varepsilon^2 |D|,$$

and hence $|D| \leq F$. □

Let $\tilde{y}_T = (\frac{1}{1+\varepsilon}) y_T$. If $e \notin D$, then $|y_T(e)| < 1 + \varepsilon$, so $|\tilde{y}_T(e)| < 1$ and $\tilde{y}_T$ has no overflow on e. If $e \in D$, then $|\mathrm{over}(\tilde{y}_T(e))| \leq |\mathrm{over}(y_T(e))|$. Hence, by the Cauchy–Schwarz inequality and Claim 34.13,

$$\|\mathrm{over}(\tilde{y}_T)\|_1 \leq \sum_{e \in D} |\mathrm{over}(y_T(e))| \leq \sqrt{|D|}\, \|\mathrm{over}(y_T)\|_2 \leq \sqrt{F} \cdot \varepsilon \sqrt{F},$$

that is,

$$\|\mathrm{over}(\tilde{y}_T)\|_1 \leq \varepsilon F.$$

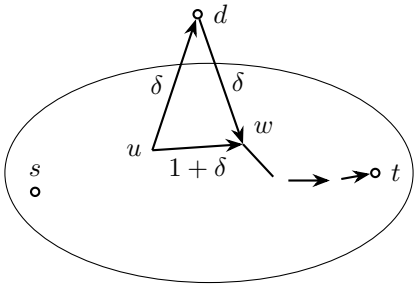
The flow $\tilde{y}_T$ has value $F/(1+\varepsilon)$. Its excess flow $\mathrm{over}(\tilde{y}_T)$ is returned to s by the procedure DrawOff, which lowers the value by at most $\|\mathrm{over}(\tilde{y}_T)\|_1 \leq \varepsilon F$. Thus the final flow has value at least

$$\begin{aligned} \frac{F}{1+\varepsilon} - \varepsilon F &= \Big(\frac{1 - \varepsilon - \varepsilon^2}{1+\varepsilon}\Big) F \\ &\geq (1 - 2\varepsilon) F, \end{aligned}$$

since $(1-2\varepsilon)(1+\varepsilon) = 1 - \varepsilon - 2\varepsilon^2 \leq 1 - \varepsilon - \varepsilon^2$. □

34.5 Drain

The procedure DrawOff starts from a flow y with overflow $\mathrm{over}(y)$, as in the picture below: an edge from u to w carries $1 + \delta$, and a vertex d carries flow δ on its edges from u and to w.



References

[ABN08] Ittai Abraham, Yair Bartal, and Ofer Neiman. Nearly tight low stretch spanning trees. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008)*, pages 781–790, 2008.

[AC07] Reid Andersen and Fan Chung. Detecting sharp drops in PageRank and a simplified local partitioning algorithm. In Jin-Yi Cai, S. Barry Cooper, and Hong Zhu, editors, *Theory and Applications of Models of Computation (TAMC 2007)*, volume 4484 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2007.

[ACL07] Reid Andersen, Fan Chung, and Kevin Lang. Using PageRank to locally partition a graph. *Internet Mathematics*, 4(1):35–64, 2007.

[Ald90] David J. Aldous. The random walk construction of uniform spanning trees and uniform labelled trees. *SIAM Journal on Discrete Mathematics*, 3(4):450–465, 1990.

[AW02] Rudolf Ahlswede and Andreas Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, 2002.

[Bro89] Andrei Broder. Generating random spanning trees. In *30th Annual Symposium on Foundations of Computer Science (FOCS 1989)*, pages 442–447, 1989.

[BSS12] Joshua Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-Ramanujan sparsifiers. *SIAM Journal on Computing*, 41(6):1704–1721, 2012.

[BT09] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[Bub15] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3–4):231–357, 2015.

[CMSV17] Michael B. Cohen, Aleksander Mądry, Piotr Sankowski, and Adrian Vladu. Negative-weight shortest paths and unit capacity minimum cost flow in $\tilde{O}(m^{10/7} \log W)$ time (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2017)*, pages 752–771, 2017.

[DBV16] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, 2016.

[FG65] Delbert R. Fulkerson and Oliver A. Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15(3):835–855, 1965.

[Fie75] Miroslav Fiedler. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, 25(4):619–633, 1975.

[GKK13] Ashish Goel, Michael Kapralov, and Sanjeev Khanna. Perfect matchings in $O(n \log n)$ time in regular bipartite graphs. *SIAM Journal on Computing*, 42(3):1392–1404, 2013.

[GW95] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[HJ12] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, second edition, 2012.

[HMR73] Alan J. Hoffman, Michael S. Martin, and Donald J. Rose. Complexity bounds for regular finite difference and finite element grids. *SIAM Journal on Numerical Analysis*, 10(2):364–369, 1973.

[JP78] D. S. Johnson and F. P. Preparata. The densest hemisphere problem. *Theoretical Computer Science*, 6(1):93–107, 1978.

[KKT95] David R. Karger, Philip N. Klein, and Robert E. Tarjan. A randomized linear-time algorithm to find minimum spanning trees. *Journal of the ACM*, 42(2):321–328, 1995.

[KMP11] Ioannis Koutis, Gary L. Miller, and Richard Peng. A nearly-$m \log n$ time solver for SDD linear systems. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS 2011)*, pages 590–598, 2011.

[KW17] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations (ICLR 2017)*, 2017.

[LLPS93] Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

[LRS13] Yin Tat Lee, Satish Rao, and Nikhil Srivastava. A new approach to computing maximum flows using electrical flows. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC 2013)*, pages 755–764, 2013.

[Nes83] Yu. E. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Doklady Akademii Nauk SSSR*, 269(3):543–547, 1983. In Russian. English translation in *Soviet Mathematics Doklady* 27 (1983), 372–376.

[Ros70] Donald J. Rose. Triangulated graphs and the elimination process. *Journal of Mathematical Analysis and Applications*, 32(3):597–609, 1970.

[Saa96] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, Boston, 1996.

[Sot15] José A. Soto. Improved analysis of a max-cut algorithm based on spectral partitioning. *SIAM Journal on Discrete Mathematics*, 29(1):259–268, 2015.

[Spi10] Daniel A. Spielman. Personal PageRank and spilling paint. Lecture notes, Graphs and Networks (Applied Mathematics / Computer Science 462/562), Lecture 11, Yale University, October 2010.

[SS11] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011.

[SW09] Daniel A. Spielman and Jaeoh Woo. A note on preconditioning by low-stretch spanning trees. arXiv:0903.2816 [cs.NA], March 2009.

[Tar79] Robert Endre Tarjan. Applications of path compression on balanced trees. *Journal of the ACM*, 26(4):690–715, 1979.

[TB97] Lloyd N. Trefethen and David Bau, III. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, 1997.

[Tre12] Luca Trevisan. Max cut and the smallest eigenvalue. *SIAM Journal on Computing*, 41(6):1769–1786, 2012.

[Tro12] Joel A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

[Vai90] Pravin M. Vaidya. Solving linear equations with symmetric diagonally dominant matrices by constructing good preconditioners. A talk based on the manuscript was presented at the IMA Workshop on Graph Theory and Sparse Matrix Computation, Minneapolis, October 1991. Unpublished manuscript, University of Illinois at Urbana-Champaign, 1990.